PI Interface for IEEE C37.118

Version 1.1.0.x

OSIsoft, LLC
777 Davis St., Suite 250
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Fax: (01) 510-357-8136
Web: http://www.osisoft.com

OSIsoft Australia • Perth, Australia
OSIsoft Europe GmbH • Frankfurt, Germany
OSIsoft Asia Pte Ltd. • Singapore
OSIsoft Canada ULC • Montreal & Calgary, Canada
OSIsoft, LLC Representative Office • Shanghai, People's Republic of China
OSIsoft Japan KK • Tokyo, Japan
OSIsoft Mexico S. De R.L. De C.V. • Mexico City, Mexico
OSIsoft do Brasil Sistemas Ltda. • Sao Paulo, Brazil
OSIsoft France EURL • Paris, France

PI Interface for IEEE C37.118

Copyright: © 2007-2015 OSIsoft, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of OSIsoft, LLC.

OSIsoft, the OSIsoft logo and logotype, PI Analytics, PI ProcessBook, PI DataLink, ProcessPoint, PI Asset Framework (PI AF), IT Monitor, MCN Health Monitor, PI System, PI ActiveView, PI ACE, PI AlarmView, PI BatchView, PI Coresight, PI Data Services, PI Event Frames, PI Manual Logger, PI ProfileView, PI WebParts, ProTRAQ, RLINK, RtAnalytics, RtBaseline, RtPortal, RtPM, RtReports and RtWebParts are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Published: 08/2015

Table of Contents

Chapter 1. Introduction 1

Reference Manuals 4
Supported Operating Systems 4
Supported Features 5
Diagram of Hardware Connection 8

Chapter 2. Principles of Operation 9

Startup Process 9
LocalEndPoint 10
RemoteEndPoint 10
PMU 10
Loading PI Points 10
Data Processing 11
Data Flow to the PI System 12
Multi-Threading 12
Communications Plug-Ins 13
PIC37118_Comm001 13
PIC37118_Comm002 13
PIC37118_Comm003 13
Data Quality Indications 13
Interface Robustness 14
PDC / PMU Data Source Level Failover 14
Failover 15
Communication Failure 15
Detection of Stale Data 15

Chapter 3. Installation Checklist 17

Data Collection Steps 17
Interface Diagnostics 18
Advanced Interface Features 19

Chapter 4. Interface Installation 21

Naming Conventions and Requirements 21
Interface Directories 22
PIHOME Directory Tree 22
Interface Installation Directory 22
Interface Installation Procedure 22
Silent Installation Procedure 22
PI Trust for Interface Authentication 22
Installing Interface as a Windows Service 23
Installing Interface Service with PI Interface Configuration Utility 23
Service Configuration 24
Installing Interface Service Manually 29

Chapter 5. Digital States **31**

Chapter 6. PointSource **33**

Chapter 7. PI Point Configuration **35**

Point Attributes 35
Tag 35
PointSource 36
PointType 36
Location1 36
Location2 36
Location3 37
Location4 37
Location5 37
InstrumentTag 37
ExDesc 40
Scan 40
Shutdown 40
DataSecurity 41
PtSecurity 41

Chapter 8. Startup Command File **43**

Configuring the Interface with PI ICU 43
C37118 Interface page 46
Sample PIC37118 XML File 63
Command-line Parameters 64
Sample PIC37118.bat File 71

Chapter 9. UniInt Failover Configuration **73**

Introduction 73
Quick Overview 74
Synchronization through a Shared File (Phase 2) 75
Configuring Synchronization through a Shared File (Phase 2) 76
Configuring UniInt Failover through a Shared File (Phase 2) 79
Start-Up Parameters 79
Failover Control Points 81
PI Points 82
Detailed Explanation of Synchronization through a Shared File (Phase 2) 86
Steady State Operation 87
Failover Configuration Using PI ICU 89
Create the Interface Instance with PI ICU 89
Configuring the UniInt Failover Startup Parameters with PI ICU 90
Creating the Failover State Digital State Set 91
Using the PI ICU Utility to create Digital State Set 91
Using the PI SMT 3 Utility to create Digital State Set 92
Creating the UniInt Failover Control and Failover State Points (Phase 2) 95

Chapter 10. Interface Node Clock **97**

Chapter 11. Security **99**

Windows 99

Authentication 99
Authorization 100

Chapter 12. Starting / Stopping the Interface 103

Starting Interface as a Service 103
Stopping Interface Running as a Service 103

Chapter 13. Buffering for PI Interfaces 105

Buffering services 105
Buffering and collectives 105
Buffering configuration 105

Chapter 14. Interface Diagnostics Configuration 107

Scan Class Performance Points 107
Performance Counters Points 110
Performance Counters 112
Performance Counters for both (_Total) and (Scan Class x) 112
Performance Counters for (_Total) only 113
Performance Counters for (Scan Class x) only 116
Interface Health Monitoring Points 117
I/O Rate Point 122

Appendix A. Error and Informational Messages 125

Message Logs 125
Messages 126
Interface Configuration Error Messages 126
Point Loading Error Messages 128
Communications Related Error Messages 129
System Errors and PI Errors 131
UniInt Failover Specific Error Messages 132
Informational 132
Errors (Phase 1 & 2) 133
Errors (Phase 2) 134

Appendix B. PI SDK Options 135

Appendix C. Communication Plug-In Selection 137

Appendix D. Terminology 139

Interface Specific Terms 139
General Terms 139

Appendix E. Technical Support and Resources 143

Appendix F. Revision History 145

Chapter 1. Introduction

This document describes the use of the PI Interface for IEEE C37.118, from here on referred to as the PI C37.118 interface. The PI C37.118 interface supports the IEEE C37.118/D7.3.5 Standard for Synchrophasors for Power Systems.

The PI C37.118 interface operates with any C37.118 compliant device via an Ethernet or serial interface. When connected to the C37.118 or Phasor Measurement Unit (PMU) or Phasor Data Concentrator (PDC) the interface provides highly accurate measurement of electrical power quality. Each instance of the interface can connect to multiple PDCs or PMUs and provides electrical measurements to the PI System.

The IEEE C37.118 specification does not specify what electrical measurements the device must provide; rather, it specifies only how the protocol-specific data (if provided) must be formatted. Therefore, each C37.118 device may provide its own unique set of Phasor measurements. The PI C37.118 interface is capable of handling specific Phasor configuration sets that are provided by the C37.118 compliant device.

The following electrical measurements and flags are typically provided by C37.118 compliant devices:

- C37.118 compliant Phasor data calculated at a rate defined by the PDC or PMU.
    - System Frequency
    - Frequency Deviation
    - Frequency Rate of Change
    - A, C, B magnitude and phase angle for Voltage and Current given in 16-bit (scaled) Polar format
    - A, C, B real and imaginary parts of complex number for Voltage and Current given in 16-bit (scaled) Rectangular format
    - A, C, B magnitude and phase angle for Voltage and Current given in 32-bit IEEE floating point Polar format
    - A, C, B real and imaginary parts of complex number for Voltage and Current given in 32-bit IEEE floating point format
    - Analog channel data as a 16-bit integer (scaled) or 32-bit IEEE floating point. The values and ranges defined by user.
    - Digital channel data as a 16-bit word with values and ranges defined by the user.

- Quality information reflecting the accuracy of the C37.118 data is available via a series of optional quality tags.
    - Data Valid quality returned in the Data Valid flag (Bit 15) of the C37.118 STAT word indicates problems with the C37.118 data block. Some data in the block may be invalid or filled in.
    - PMU Error quality returned in the PMU Error flag (Bit 14) of the C37.118 STAT word indicates internal errors with the PMU such as A/D calibration errors, memory errors, etc.
    - PMU Sync Error quality returned in the PMU Sync flag (Bit 13) of the C37.118 word indicates a loss of external time synchronization such as the loss of satellite tracking or IRIG-B input failure.
    - Data Sorting information used by PDC's to indicate the method by which the PMU data is inserted into the data frame. This information is extracted from the Data Sorting flag (Bit 12) of the C37.118 STAT word.
    - Time Quality indicator codes extracted from the Time Quality field (Bits 3-0) of the C37.118 FRACSEC word can be used with the PMU Sync Error quality and indicates the accuracy of the timestamp provided.
    - Time Lock quality can be used in conjunction with the PMU Sync Error quality to determine the length of time that time sync has been lost. This information is extracted from the Unlocked Time flags (Bits 4 and 5) of the C37.118 STAT word.
    - Leap Second status can be extracted from the Leap Second flags (Bits 6-4) of the C37.118 FRACSEC word indicates if a leap second is pending or occurring and indicates the direction of the leap.
- Non C37.118 measurement information can also be stored in the PI Data Archive.
    - Interface diagnostic information
    - Longitude and Latitude of the PMU.

Measured data from the PMU is time-stamped by the PMU in accordance with the C37.118 standard for C37.118 data. Timestamp accuracy is achieved through an internal Global Positioning System (GPS) satellite receiver that synchronizes the PMU to within 1 µs of Coordinated Universal Time (UTC).

If a PDC or PMU indicates a data quality or time synchronization issue by setting any of the Data Valid, PMU Error, PMU Sync Error or Data Sorting Type flags in the STAT word, the interface can do one of the following (this behavior is controlled via the XML configuration file):

- Store normally.
- Mark all PMU measurements as Questionable.
- Apply a System Digital State corresponding to the flag(s) that are set. If multiple flags are set, the most critical applies.
- Discard the measurement values.

Multiple instances of the interface can be run on a single PI interface node. However, when running multiple instances of the interface on a single node, CPU and memory allocation should be monitored to ensure the system remains in a stable state of operation. The exact number of instances that may be run on a single PI interface node will vary according to hardware and network specifications.

Both PDC / PMU Data Source Level Failover and UniInt Phase 2 interface level failover are supported. When running in PDC / PMU Data Source Level Failover mode, the interface obtains data from one of two available sources PDC / PMU. The source PDC / PMU must have identical phasors, analog and digital channels and data streams for each interface. This requirement ensures that the interface obtains the same data regardless of which source is active. When running in UniInt Failover mode, two copies of the interface are connected to the PMU/PDC at the same time. When the primary interface stops collecting data, the backup interface assumes the primary role and continues data collection. PMU/PDC Data Source Level Failover and UniInt Phase 2 interface level failover modes can be run simultaneously. Failover maximizes interface data availability on the receiving PI Data Archive(s).

Note: It should be noted that the C37.118 specification is very vague with respect to communications configuration, and the various PDC / PMU vendors have implemented devices in a variety of ways. As such, the design of the C37118 interface uses an XML configuration file to provide the information needed to establish communications with the various C37.118 devices. **Check with OSIsoft to verify that your make and model of PDC or PMU has been validated for use with the C37118 interface.**

Note: The value of the [PIHOME] variable for the 32-bit interface will depend on whether the interface is being installed on a 32-bit operating system (`C:\Program Files\PIPC`) or a 64-bit operating system (`C:\Program Files (x86)\PIPC`).

The value of the [PIHOME64] variable for a 64-bit interface will be: `C:\Program Files\PIPC` on a 64-bit operating system.

In this documentation [PIHOME] is used to represent the value for either [PIHOME] or [PIHOME64]. The value of [PIHOME] is the directory in which PI client applications are stored.

Note: This interface has been built against a UniInt version (4.5.0.59 and later) which now writes all its messages to the local PI Message log.

Please see the document *UniInt Interface Message Logging.docx* in the `%PIHOME%\Interfaces\UniInt` directory for more details about how to access these messages.

Note: OSIsoft is revising product documentation and other literature to reflect the evolution of the PI Server from a single server to a multi-server architecture. Specifically, the original historian core of the PI Server is now referred to as the PI Data Archive.

Originally, the PI Server was a single server that contained the PI Data Archive and other subsystems. To add features and improve scalability, the PI Server has evolved from a single server to multiple servers. While the PI Data Archive remains a core server of the PI Server product, the product name "PI Server" now refers to much more than the PI Data Archive. OSIsoft documentation, including this user manual, is changing to use "PI Server" in this broader sense and "PI Data Archive" to refer to the historian core.

Reference Manuals

OSIsoft

- PI Data Archive manuals
- *PI API Installation Instructions* manual (%PIHOME%\bin\API_install.doc)
- *PI Universal Interface (UniInt) User Guide* (%PIHOME%\Interfaces\UniInt\PI Universal Interface (UniInt) User Guide.pdf)
- *UniInt Interface Message Logging for UniInt 4.5.0.x and later Interfaces* (%PIHOME%\Interfaces\UniInt\UniInt Interface Message Logging.pdf)
- *PI Interface Configuration Utility* User Guide (%PIHOME%\ICU\PI Interface Configuration Utility.pdf)

Vendor

- *IEEE C37.118/D7.3.5 Standard for Synchrophasors for Power Systems*

Supported Operating Systems

Platforms		32-bit application	64-bit application
Windows Vista	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 2008	32-bit OS	Yes	No
Windows 2008 R2	64-bit OS	Yes (Emulation Mode)	No
Windows 7	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 8	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 2012 Server	64-bit OS	Yes (Emulation Mode)	No

The interface is designed to run on the above mentioned Microsoft Windows operating systems and their associated service packs.

Please contact OSIsoft Technical Support for more information.

Security Note: OSIsoft recommends installing all available updates from the Windows Update service. OSIsoft also recommends using the newest versions of Windows to obtain the latest security features.

Supported Features

Feature	Support
Interface Part Number	PI-IN-OS-C37.118-NTI
Auto Creates PI Points	No
Point Builder Utility	No
ICU Control	Yes
PI Point Types	Digital / Int16 / Int32 / Float16 / Float32
Sub-second Timestamps	Yes
Sub-second Scan Classes	No (Unsolicited Only)
Automatically Incorporates PI Point Attribute Changes	Yes
Exception Reporting	No
Outputs from PI	No
Inputs to PI:	Unsolicited
Supports Questionable Bit	Yes
Supports Multi-character PointSource	Yes
Maximum Point Count	No
* Uses PI SDK	No
PINet String Support	No
* Source of Timestamps	C37.118 Datastream
History Recovery	No
* UniInt-based * Disconnected Startup * SetDeviceStatus	Yes Yes Yes
* Failover	Data Source Level Failover UniInt Failover (Phase 2 Cold and Hot)
* Vendor Software Required on Interface Node / PINet Node	No
Vendor Software Required on Foreign Device	No
Vendor Hardware Required	Yes
Additional PI Software Included with interface	No
Device Point Types	Digital, Float, Integer (including Real and Imaginary)
Serial-Based interface	Yes

** See paragraphs below for further explanation.*

Uses PI SDK

The PI SDK and the PI API are bundled together and must be installed on each interface node. This interface does not specifically make PI SDK calls.

Source of Timestamps

Timestamps are acquired from the C37.118 data stream.

UniInt-based

UniInt stands for Universal Interface. UniInt is not a separate product or file; it is an OSIsoft-developed template used by developers and is integrated into many interfaces, including this interface. The purpose of UniInt is to keep a consistent feature set and behavior across as many OSIsoft interfaces as possible. It also allows for the very rapid development of new interfaces. In any UniInt-based interface, the interface uses some of the UniInt-supplied configuration parameters and some interface-specific parameters. UniInt is constantly being upgraded with new options and features.

The *PI Universal Interface (UniInt) User Guide* is a supplement to this manual.

Disconnected Start-Up

The PI IEEE C37.118 interface is built with a version of UniInt that supports disconnected start-up. Disconnected start-up is the ability to start the interface without a connection to the PI Data Archive. This functionality is enabled by adding **`/cachemode`** to the list of start-up parameters or by enabling disconnected startup using the ICU. Refer to the *PI Universal Interface (UniInt) User Guide* for more details about UniInt Disconnect startup.

SetDeviceStatus

The health point, having the attribute Exdesc = [UI_DEVSTAT], is used to represent the status of the source device. The following events can be written into this point:

- "1 | Starting | UI #.#.#.#" – The interface is currently starting up.
- "Good" – The interface is properly communicating and reading data from all configured PMUs. A value of "Good" does not mean that all points are receiving good values, but it is a good indication that there are no hardware or network problems.
- "3 | 1 devices(s) in error | REP: 1 LEP 1" – The connection to one or more PMUs has failed.
- "4 | Intf Shutdown" – The interface has been shutdown.

Refer to the *PI Universal Interface (UniInt) User Guide* for more information on how to configure health points.

Failover

- PDC / PMU Data Source Level Failover
    - PDC / PMU Data Source Level Failover maximizes interface data availability on the receiving PI Data Archive(s). It enables the interface to obtain data from one of two PMU/PDC data sources. Each PMU/PDC data source must have identical tag definitions and data streams for interface source points.
    - The interface initiates failover if the active source data becomes stale or is not available due to network issues.
- UniInt Failover Support

UniInt Phase 2 Failover provides support for cold, warm or hot failover configurations. The Phase 2 hot failover results in a *no data loss* solution for bi-directional data transfer between the PI Data Archive and the Data Source given a single point of failure in the system architecture similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition. This failover solution requires that two copies of the interface be installed on different interface nodes, both of which are collecting data simultaneously from a single data source. Phase 2 Failover requires that each interface have access to a shared data file. Failover operation is automatic and operates with no user interaction. Each interface participating in failover has the ability to monitor and determine liveliness and failover status. To assist in administering system operations, the ability to manually trigger failover to a desired interface is also supported by the failover scheme.

The failover scheme is described in detail in the *PI Universal Interface (UniInt) User Guide*, which is a supplement to this manual. Details for configuring this interface to use failover are described in the UniInt Failover Configuration section of this manual.

This interface supports UniInt Failover (Phase 2 Cold and Hot).

Vendor Hardware Required

A C37.118-compliant Phasor Measurement Unit or Phasor Data Concentrator is required for proper communication with this interface.

Device Point Types

The interface supports all Phasor, Analog and Digital channels plus frequency and frequency rate of change. Frequency error is also computed by the interface.

Serial-Based Interface

This interface can run either with a serial connection or Ethernet network connection.

Server class machines often have low-quality serial ports. Server class machines are not required for most interfaces and should not be used, especially when serial port connections are required.

Diagram of Hardware Connection

Chapter 2. Principles of Operation

The PI C37.118 interface is designed to request and process data from C37.118 compliant devices. The PI C37.118 interface is a real-time based interface. The interface operations can be categorized into the following list of sub-operations:

- Startup Process
- Loading PI Points
- Data Processing
- Data Flow to the PI System
- Data Quality Indicators
- Interface Robustness

A more detailed explanation of the list of sub-operations for this interface is provided in this section.

Startup Process

The PI C37.118 interface must be configured in accordance with the Startup Command File (`.bat` file) and the Interface Configuration File (XML configuration file). Each instance of the C37118 interface is assigned a specific identification number and/or point source, which are correlated with the interface-specific startup file parameters **`/id`**, **`/ps`** and **`/configfile`**.

The C37118 interface establishes the initial connection to PI and reconnects to PI in the event that the connection is lost. If the interface is started while the PI Data Archive is down, the interface periodically attempt to establish a connection until the PI Data Archive is up and running.

The interface is designed to make use of a configuration file to set up the internal operation of the interface. The configuration file is written in XML format. A PI Interface Configuration Utility (ICU) plug-in is available for this interface to assist in the configuration of both the startup batch file and the interface (XML) configuration file. The use of this ICU Control (which is highly recommended) is described in the Startup Command File section of this document.

When the interface starts, it parses the command line parameters and searches for the XML configuration file that is specified in the **`/configfile`** command line parameter. If the **`/configfile=<UNCpath>`** is not found, a default file name (`pic37118.xml`), is assumed. The configuration file is then parsed and the interface instantiates the internal structures that are required for communication with the C37.118 PDC and / or PMUs.

A description of the internal structures and the terms used by the interface are described below.

LocalEndPoint

The LocalEndPoint represents the local (interface side) of a physical network or serial connection to a C37.118 device. Each LocalEndPoint should contain the various LocalEndPoint attributes as well as a single RemoteEndPoint leaf node which contains the RemoteEndPoint attributes.

RemoteEndPoint

The RemoteEndPoint object represents the remote (C37.118 device) side of a physical connection. Currently there can be only one RemoteEndPoint configured for each LocalEndPoint.

PMU

The PMU object represents a C37.118 Phasor Data Concentrator (PDC) or Phasor Measurement Unit (PMU). If the RemoteEndPoint is a PDC, there can be multiple PMU objects associated with the RemoteEndPoint.

Loading PI Points

When the PI C37.118 interface starts, it searches the PI Point Database for points that belong to the interface and creates a point list for the interface. The primary method of establishing a mapping between a PI point and a C37.118 device is through configuration of the `Location2`, `Location3`, `Location5`, and `InstrumentTag` PI point attributes. The correlation between the above attributes and the various C37.118 measurements are listed in the PI Point Configuration section of this document.

PI points for this interface have a one-to-one correspondence with electrical measurements that are available from the C37.118 device. There can be only one PI point for receiving a specific measurement from a given C37.118 device.

For example: if a PI point is configured to read the magnitude from a Phasor channel named "Phasor Volt CH-A" (`InstrumentTag=PhasorMag\Phasor Volt CH-A`) from a specific LocalEndPoint, RemoteEndPoint and PMU combination (**`Location2`**, **`Location3`** and **`Location5`**) there can be no other PI points specified with identical attributes.

After the interface loads a PI point, any additional PI points that have the same attributes as a previously loaded PI point are rejected by the interface and the interface logs an error message. Refer to Appendix A. Error and Informational Messages for more details about messages that are associated with loading PI points.

Data Processing

The PI C37.118 interface processes C37.118 formatted data messages as defined in the IEEE C37.118 specification. The input data is parsed to provide the following:

- Real and Imaginary part of the complex number for each Phasor channel if the PDC / PMU is configured to send Phasor measurements in rectangular format.
- Magnitude and Phase Angle for each Phasor channel for each Phasor channel if the PDC / PMU is configured to send Phasor measurements in polar format.
- Analog channel data
- Digital channel data
- System Frequency
- Frequency deviation (calculated by interface)
- Frequency Rate of Change

Note: If the PMU is configured to send data in rectangular format, the interface computes the polar coordinates; if the PMU is configured to send data in polar format the interface computes the rectangular coordinates.

In addition to the data parsed from the C37.118 data, the interface also provides for input of C37.118 configuration data from the CONFIG2 data block. The configuration data is parsed to provide the following:

- Configuration change count from CFGCNT
- Data transmit rate from DATA_RATE
- Nominal frequency (50Hz / 60Hz)

Optional data quality tags can be configured to receive information related to the quality of the C37.118 data. This quality information is extracted from the time quality, leap second status, and STAT fields of the C37.118 data stream. These include the following:

- Data Valid
- PMU Error
- PMU Sync
- Data Sorting
- Time Unlocked Time
- Trigger Reason

The interface can also store geographical information regarding the location of the PMU if defined in the Interface Configuration File. The Latitude and Longitude of the PMU can be defined in decimal form in the configuration file and stored in the PI Data Archive.

Data Flow to the PI System

Data flow from the C37.118 device to the PI System is unsolicited. The following is an explanation of the sequence of events that take place in order to transfer data from the C37.118 device interface into the PI System.

Phasor data from C37.118 is sent to the interface at a rate which is configured in the C37.118 device and is typically 20, 30, 60 or 120 times per second for 60Hz systems and 25, 50, and 100 times per seconds for 50Hz systems.

When the interface communications plug-in issues the start data command (which is issued at startup and after any communications failure), the C37.118 device starts sending data at the configured rate.

The communications plug-in then processes the incoming data stream. Each C37.118 data frame is processed and the size and CRC verified. The C37.118 frame is then queued to the main interface thread.

The main interface thread of the interface parses and processes C37.118 data frames in the incoming data queue and sends the results to the PI System on a real-time basis. This means that device data, after processing, is sent immediately and is available in the PI System.

The illustration below shows a graphical representation of data flow from the C37.118 PDC / PMU device to the PI System.

Multi-Threading

The PI C37.118 interface is multithreaded. Communications with the PDC / PMU is handled in separate threads from the main interface body. Depending on the Communications Plug-In being used, there will be either 1 or 2 communications threads created. If commands and command responses operate on the same communications port as the C37.118 data stream, a single communications thread will handle all communications with the PDC / PMU. If commands and command responses operate on independent ports then 2 communications threads are created. One thread handles commands and command responses and the second is dedicated to receiving the C37.118 data stream. Data from each communications thread is buffered and then serialized through the main interface thread. This must be done because the PI API is not currently thread safe.

Communications Plug-Ins

The PI C37.118 interface uses “Communications Plug-Ins” to handle differences in the application layer communications for various PDCs and PMUs. This is necessary because the IEEE C37.118 specification does not dictate the implementation.

The following describes the plug-ins distributed with the PI C37.118 interface.

PIC37118_Comm001

This plug-in is used for devices that use a single socket via either UDP or TCP for commands, command responses, and measurement data. This plug-in is also used for serial connections.

PIC37118_Comm002

This plug-in is used for devices that use two sockets. Commands and command responses are performed via one socket and measurement data is returned via a separate socket. Typically the device uses TCP for the commands and UDP for the measurement data.

PIC37118_Comm003

This plug-in is used for devices that use a single socket via either UDP or TCP but do not support output commands. Once the device starts it sends data automatically to a predetermined address and port. This is referred to as “Autonomous Mode” on most devices. It expects that the CONFIG2 block is sent periodically along with the measurement data.

Data Quality Indications

The PI C37.118 interface provides multiple methods to monitor the quality of the data being provided by the C37.118 PDC / PMU.

The following types of data quality indications are available. All of these are optional and can be used alone or in conjunction with other data quality indicators.

- Quality Tags – Quality tags are PI tags that are separate from data tags.
- Conditional Data Tag Status – The interface can be configured to apply one of 4 different rules in response to one or more of the data quality flags being set.

Interface Robustness

Where data acquisition is concerned, the C37118 interface is designed to be as robust as possible. If the interface determines that Phasor data has not been received within the time specified by the **`DataTimeout`** parameter in the configuration file, an entry is made to the *pipc.log* file and the interface attempts to restart the C37.118 phasor data stream. Upon receipt of a successful response from the C37.118 device, the interface begins processing data and logs a message to the `pipc.log` file. The interface will retry failed connection attempts until data flow is restored but will not log further connection failure messages until data flow is restored. Refer to Appendix A. Error and Informational Messages for more details about messages that are associated with transmission loss.

The interface is configured to make use of the **`WriteIOTimeout`** parameter in the interface configuration file. This parameter instructs the interface about whether to write an error state to PI points when data from the device is not received after the period specified by the **`DataTimeout`** parameters. Setting **`WriteIOTimeout=1`** (the default), causes the system digital state of **`"I/O Timeout"`** to be written to PI points. Alternately, setting **`WriteIOTimeout=0`** instructs the interface not to update PI points during transmission loss of C37.118 data. The default behavior is to write "I/O Timeout" for all measured values.

When the interface starts, the interface searches the PI Point Database for points that belong to the interface and a point list is created for the interface.

When startup is complete, the interface enters the processing loop, which includes the following:

- Servicing scheduled input points. Each Scan Class is processed in turn.
- Servicing triggered input points as events arrive.
- The PI Point Database is checked every 2 minutes for points that are added, edited, and deleted. If point updates are detected, the points are loaded (or reloaded) by the interface as appropriate. The 2-minute update interval can be adjusted with the **`/updateinterval`** command-line parameter described in the *PI Universal Interface (UniInt) User Guide*. The interface will only process 25 point updates at a time. If more than 25 points are added, edited, or deleted at one time, the interface will process the first 25 points, wait 30 seconds (or by the time specified by the **`/updateinterval`** parameter, whichever is lower), process the next 25 points, and so on. After all points have been processed, the interface resumes checking for updates every 2 minutes (or by the time specified by the **`/updateinterval`** parameter). The interface writes the digital state `SCAN OFF` to any points that are removed from the interface while it is running.

PDC / PMU Data Source Level Failover

PDC / PMU Data Source Level Failover maximizes interface data availability on the receiving C37.118 data stream. It requires that two PDC / PMU data sources are available that have identical Phasor, analog and digital channels and data streams for interface points. This requirement ensures that the interface obtains the same data regardless of which data source is active.

Failover is enabled by specifying a pair of primary and secondary Local End Points by specifying Local End Point and Failover Local End Pont IDs in the xml configuration file. On startup, the interface attempts to establish a connection to the primary data source, then load and initialize its tag list. Data collection begins from the primary PDC / PMU data source making it the active node while the secondary PDC / PMU data source assumes the standby (backup) role.

To set up PDC / PMU Data Source Level Failover you must create at least two Local End Points. You should bind the end points together to create a redundant pair. To do so, use ID and Failover ID fields. Failover ID should be equal to the ID of the redundant Local End Point and vice versa. Only one Local End Point in the redundant pair should be specified as a primary. Use the Primary field to specify which Local End Point is a primary. Another one will be a secondary Local End Point.

To control delay between switches from primary to backup PDC /PMU use the Switch Backup Delay parameter. Default value is 60 seconds.

You can have multiple redundant pairs configured for the interface, or you can set up redundant Local End Points and non-redundant (Failover ID field not assigned) together for the same interface.

Failover

There are two conditions that cause the interface to initiate failover: a communication failure or detection of stale data.

Communication Failure

The first failover condition occurs when the interface loses communication to the active PDC / PMU data source. This might be the result of a temporary network outage, shutdown of the active data source, hardware failure, and so on. The interface cannot identify the cause of the the disconnect failure. A disconnection means that the interface did not receive a response from the active source within the timeout period. The interface may initiate a short wait and then attempt to reconnect to the active source before attempting failover. The data timeout, reconnection rate, and reconnection timeout are all user configurable parameters.

The default reconnection timeout period is 60 seconds, which is set as an interface configuration parameter. If the interface is unable to reconnect to the active PDC / PMU during the reconnection timeout period, the interface attempts to connect to the alternate (primary or backup) PDC / PMU.

Detection of Stale Data

The second failover condition is caused by stale source data. The data received by the PDC / PMU is high frequency. Depending on the brand and configuration of each individual PMU data might arrive at a rate of 10, 20, 30 or 60 Hz. If the interface does not receive data during the data timeout interval (which defaults to 1000 milliseconds) the data is marked as timed out and becomes stale. The interface attempts to reconnect with the active PDC / PMU at the period specified by the reconnection rate interval. After the reconnection timeout interval is expired, the interface attempts to connect to the alternate (backup or primary) PDC /PMU. After each reconnection timeout interval the interface attempts to switch back and forth

between the primary and the backup PDC /PMU until it receives data from the first available PDC / PMU.

UniInt Failover

This interface supports UniInt failover. Refer to the UniInt Failover Configuration section of this document for configuring the interface for failover.

Chapter 3. Installation Checklist

If you are familiar with running PI data collection interface programs, this checklist helps you get the interface running. If you are not familiar with PI interfaces, return to this section after reading the rest of the manual in detail.

This checklist summarizes the steps for installing this interface. You need not perform a given task if you have already done so as part of the installation of another interface. For example, you only have to configure one instance of Buffering for every interface node regardless of how many interfaces run on that node.

The Data Collection Steps below are required. Interface Diagnostics and Advanced Interface Features are optional.

Data Collection Steps

1. Confirm that you are able to use PI SMT to configure the PI Data Archive. You need not run PI SMT on the same computer on which you run this interface.
2. If you are running the interface on an interface node, edit the PI Data Archive's Trust Table to allow the interface to read attributes and point data. If a buffering application is not running on the interface node, the PI trust must allow the interface to write data.
3. Run the installation kit for the PI Interface Configuration Utility (ICU) on the interface node if the ICU will be used to configure the interface. This kit runs the PI SDK installation kit, which installs both the PI API and the PI SDK.
4. Run the installation kit for this interface. This kit also runs the PI SDK installation kit which installs both the PI API and the PI SDK if necessary.
5. If you are running the interface on an interface node, check the computer's time zone properties. An improper time zone configuration can cause the PI Data Archive to reject the data that this interface writes.
6. Run the ICU and configure a new instance of this interface. Essential startup parameters for this interface are:

   Point Source (**/PS=***x*)
   Interface ID (**/ID=***#*)
   PI Data Archive (**/Host=***host:port*)
7. If you will use digital points, define the appropriate digital state sets.

8. Build input tags for this interface. Important point attributes and their purposes are:

   Location1 specifies the interface instance ID.
   Location2 specifies the LocalEndPoint ID.
   Location3 specifies the RemoteEndPoint ID. This attribute always reflects the IDCODE of PDC / PMU contained in word 3 of the C37.118 data stream. If the connected device is a PMU (versus a PDC), Location3 and Location5 will be the same.
   Location4 is not used.
   Location5 specifies the PMU ID. This attribute will always match the IDCODE of the PMU that is the source of the data.
   ExDesc is not used.
   InstrumentTag specifies the specific measurement and type.

   PtSecurity must permit read access for the PI identity, group, or user configured in the PI trust that is used by the interface.

   DataSecurity must permit read access (buffering enabled) or read/write access (unbuffered) for the PI identity, group, or user configured in the PI trust that is used by the interface.

   Security Note: When buffering is configured, the DataSecurity attribute must permit write access for the *buffering application's* PI trust or mapping. DataSecurity write permission for the interface's PI trust is required only when buffering is not configured.

9. Start the interface interactively and confirm its successful connection to the PI Data Archive without buffering. (The DataSecurity attribute for interface points must permit write access for the interface's PI trust.)
10. Confirm that the interface collects data successfully.
11. Stop the interface and configure a buffering application (either Bufserv or PIBufss).
12. Start the buffering application and the interface. Confirm that the interface works together with the buffering application by physically removing the connection between the interface node and the PI Data Archive node. (The DataSecurity attribute for interface points must permit write access for the *buffering application's* PI trust or mapping. The interface's PI trust does not require DataSecurity write permission.)
13. Configure the interface to run as a Windows service. Confirm that the interface runs properly as a service.
14. Restart the interface node and confirm that the interface and the buffering application restart.

Interface Diagnostics

1. Configure scan class performance points.
2. Install the PI Interface for Performance Monitor on the interface node.
3. Configure performance counter points.
4. Configure UniInt health monitoring points
5. Configure the I/O Rate point.

Advanced Interface Features

1. Configure the interface for disconnected startup. Refer to the *PI Universal Interface (UniInt) User Guide* for more details on UniInt disconnected startup.
2. Configure UniInt failover; see the UniInt Failover Configuration chapter in this document for details related to configuring the interface for failover.

Chapter 4. Interface Installation

OSIsoft recommends that interfaces be installed on PI interface nodes instead of directly on the PI Data Archive node. A PI interface node is any node other than the PI Data Archive node where the PI Application Programming Interface (PI API) is installed (see the PI API manual). With this approach, the PI Data Archive need not compete with interfaces for the machine's resources. The primary function of the PI Data Archive is to archive data and to service clients that request data.

After the interface has been installed and tested, Buffering should be enabled on the PI interface node. Buffering refers to either PI API Buffer Server (Bufserv) or the PI Buffer Subsystem (PIBufss). For more information about Buffering see the Buffering for PI Interfaces chapter of this manual.

In most cases, interfaces on PI interface nodes should be installed as automatic services. Services keep running after the user logs off. Automatic services automatically restart when the computer is restarted, which is useful in the event of a power failure.

The guidelines are different if an interface is installed on the PI Data Archive node. In this case, the typical procedure is to install the PI Data Archive as an automatic service and install the interface as an automatic service that depends on the PI Update Manager and PI Network Manager services. This typical scenario assumes that Buffering is not enabled on the PI Data Archive node. Bufserv can be enabled on the PI Data Archive node so that interfaces on the PI Data Archive node do not need to be started and stopped in conjunction with PI, but it is not standard practice to enable buffering on the PI Data Archive node. The PI Buffer Subsystem can also be installed on the PI Data Archive.

Naming Conventions and Requirements

In the installation procedure below, it is assumed that the name of the interface executable is `PIC37118.exe` and that the startup command file is called `PIC37118.bat`.

When Configuring the Interface Manually

It is customary for the user to rename the executable and the startup command file when multiple copies of the interface are run. For example, `PIC371181.exe` and `PIC371181.bat` would typically be used for interface number 1, `PIC371182.exe` and `PIC371182.bat` for interface number 2, and so on. When an interface is run as a service, the executable and the command file must have the same root name because the service looks for its command-line parameters in a file that has the same root name.

Interface Directories

PIHOME Directory Tree

The [PIHOME] directory tree is defined by the PIHOME entry in the pipc.ini configuration file. This pipc.ini file is an ASCII text file, which is located in the %windir% directory.

For 32-bit operating systems, a typical pipc.ini file contains the following lines:

```
[PIPC]
PIHOME=C:\Program Files\PIPC
```

For 64-bit operating systems, a typical pipc.ini file contains the following lines:

```
[PIPC]
PIHOME=C:\Program Files (X86)\PIPC
```

The above lines define the root of the PIHOME directory on the C: drive. The PIHOME directory does not need to be on the C: drive. OSIsoft recommends using the paths shown above as the root PIHOME directory name.

Interface Installation Directory

The interface install kit will automatically install the interface to:

PIHOME\Interfaces\C37118\

PIHOME is defined in the pipc.ini file.

Interface Installation Procedure

The C37118 interface setup program uses the services of the Microsoft Windows Installer. Windows Installer is a standard part of Windows operating systems. To install, run the appropriate installation kit.

```
C37118_#.#.#.#_.exe
```

Silent Installation Procedure

To launch a silent installation, type:

```
Setup.exe -f silent.ini
```

The silent.ini file is included in the interface installation kit. You can make site-specific alterations to the file as needed. See the silent.ini file for further information and descriptions of available arguments.

PI Trust for Interface Authentication

A PI Interface usually runs on an interface node as a Windows service, which is a non-interactive environment. In order for an interface to authenticate itself to a PI Data Archive and obtain the access permissions for proper operation, the PI Data Archive must have a PI trust that matches the connection credentials of the interface. Determine if a suitable

PI trust for the interface exists on the PI Data Archive. If a suitable PI trust does not exist, see the Security chapter for instructions on creating a new PI trust.

Installing Interface as a Windows Service

The preferred way to install the C37118 interface service is to use the PI Interface Configuration Utility; however, you can also create it manually.

Security Note: For improved security, OSIsot recommends running the interface service under a non-administrative account, such as a Windows built-in service virtual account, the built-in Network Service account, or a non-administrative account that you create.

Installing Interface Service with PI Interface Configuration Utility

The PI Interface Configuration Utility provides a user interface for creating, editing, and deleting the interface service:

Service Configuration

Service name

The *Service name* box shows the name of the current interface service. This service name is obtained from the interface executable.

ID

This is the service ID that is used to distinguish among multiple instances of the same interface using the same executable.

Display name

The *Display name* text box shows the current Display Name of the interface service. If there is currently no service for the selected interface, the default Display Name is the service name with a "PI-" prefix. Users may specify a different Display Name. OSIsoft suggests that the prefix "PI-" be appended to the beginning of the interface name to indicate that the service is part of the OSIsoft suite of products.

Log on as

The *Log on as:* options allow the interface service to be configured to use a *default Windows Service* account, a *user-specified Windows* account or the *Local System* account.

Default Windows Service account

Select the ***NT Service\PIC371181*** option to configure the internet service to use a *default Windows Service* account.

This figure shows the option to select to configure the interface service to use a *default Windows Service* account.

User-specified Windows account

Select the ***[Domain\]UserName*** option to add a ***user-specified Windows*** account which the interface service will use. Please note that this account differs from the *default Windows Service* account.

This figure shows the option to select to configure the interface service to use a *user-specifed Windows* account.

When the ***[Domain\]UserName*** option is selected, three additional boxes are displayed. These additional boxes are used to enter information about the ***user-defined Windows*** account.

This figure shows an example of how to populate the configuration boxes to allow the interface service to use a *user-specifed Windows* account.

- UserName

  The *UserName:* box allows the entry of the Windows User account to be used by the interface service.

- Password

  If a Windows User account is entered into the *UserName:* box, the password for the entered account must be provided in the *Password:* box, unless the account requires no password.

- Confirm password

  If a password is entered into the *Password:* box, it must be confirmed in the *Confirm Password:* box.

Local System account

Select the ***[Domain\]UserName*** option and populate the *UserName* box with the value *LocalSystem* to configure the interface service as a *Local System* account. Leave the *Password:* and *Confirm password:* boxes empty.

This figure shows an example of how to populate the configuration boxes to configure the interface service to run as a *Local System* account.

After the interface service has been created to use the *LocalSystem* account, the ***Service Configuration / Log on as:*** area is redisplayed with a different dialog view.

This figure shows how the *Service Configuration / Log on as:* area is redrawn when the interface service is configured to use the *Local System* account.

This figure shows the interface after it has been created to use the *Local System* account.

Security Note: For best security, we recommend running this interface service under an account with minimum privileges, such as a Windows built-in service virtual account, the built-in Network Service account, or a non administrative account that you create.

PI ICU versions earlier than 1.4.14.x cannot create a service that runs as a Windows built-in service virtual account or the built-in Network Service or Local Service accounts. After ICU creates the interface service, you can change the account with a Windows administrative tool, such as *Services on the Control Panel* or the *sc* command-line utility.

Dependencies

The *Installed services* list is a list of the services currently installed on this machine.

Services upon which this interface is dependent should be moved into the *Dependencies:* list using the ***Add*** button, . For example, if API Buffering is enabled, then *Bufserv* or *PIBufss* should be selected from the list at the right and added to the list on the left. To remove a service from the list of dependencies, use the ***Remove*** button, , and the service name will be removed from the *Dependencies:* list.

When the interface is started (as a service), the services listed in the dependency list will be verified as running (or an attempt will be made to start them). If the dependent service(s) cannot be started for any reason, then the interface service will not run.

Note: Please see the PI Log and Windows Event Logger for messages that may indicate the cause for any service not running as expected.

- *Add* Button

To add a dependency from the list of *Installed services:*, select the dependency name and click the *Add* button.

- *Remove* Button

To remove a dependency from the list of *Dependencies:*, highlight the service name in the *Dependencies* list and click the *Remove* button.

The full name of the service selected in the *Installed services:* list is displayed below the *Installed services:* list box.

Startup Type

The *Startup Type* indicates whether the interface service will start automatically or must be started manually upon reboot.

- If the *Auto* option is selected, the service will be installed to start automatically when the machine reboots.
- If the *Manual* option is selected, the interface service will not start on reboot, but will require manual intervention to start the service.

- If the *Disabled* option is selected, the service will not start at all.

Generally, interface services are set to start automatically.

Create

The *Create* button adds the displayed service with the specified *Dependencies* and with the specified *Startup Type*.

Remove

The *Remove* button removes the displayed service. If the service is not currently installed, or if the service is currently running, this button will be grayed out.

Start or Stop Service

The toolbar contains a *Start* button ▶ and a *Stop* button ■. If this interface service is not currently installed, these buttons will remain grayed out until the service is added. If this interface service is running, the *Stop* button is available. If this service is not running, the *Start* button is available.

The status of the interface service is indicated in the lower portion of the PI ICU dialog.

Installing Interface Service Manually

Help for installing the interface as a service is available at any time with the command:

```
PIC37118.exe /help
```

Open a Windows command prompt window and change to the directory where the `PIC371181.exe` executable is located. Then, consult the following table to determine the appropriate service installation command.

Note: In the following Windows Service Installation Commands you may use either a slash (/) or dash (-) as the delimiter.

Windows Service Installation Commands on an Interface Node or a PI Data Archive Node with Bufserv implemented	
Manual service	PIC37118.exe /install /depend “tcpip bufserv”
Automatic service	PIC37118.exe /install /auto /depend “tcpip bufserv”
*Automatic service with service ID	PIC37118.exe /serviceid *X* /install /auto /depend “tcpip bufserv”
Windows Service Installation Commands on an Interface Node or a PI Data Archive Node without Bufserv implemented	
Manual service	PIC37118.exe /install /depend tcpip
Automatic service	PIC37118.exe /install /auto /depend tcpip
*Automatic service with service ID	PIC37118.exe /serviceid *X* /install /auto /depend tcpip

*When specifying service ID, the user must include an ID number. It is suggested that this number correspond to the interface ID (**`/id`**) parameter found in the interface .bat file.

Check the Microsoft Windows Services control panel to verify that the service was added successfully. The services control panel can be used at any time to change the interface from an automatic service to a manual service or vice versa.

The service installation commands in this section create an interface service that runs under a low-privilege built-in account. On Windows 7 and Server 2012 and later, the service logs on as the service virtual account. For earlier versions of Windows, the service logs on as Network Service.

Security Note: For best security, we recommend running this interface service under an account with minimum privileges, such as a Windows service virtual account, the built-in Network Service account, or a non administrative account that you create.

As discussed earlier, following the recommendation to run the interface service under a low-privilege account may affect performance counters.

The services control panel can change the account that the interface service runs under. Changing the account while the interface service is running does not take effect until the interface service is restarted.

Chapter 5. Digital States

For more information regarding Digital States, refer to the PI Data Archive documentation.

Digital State Sets

PI digital states are discrete values represented by strings. These strings are organized in the PI Data Archive as digital state sets. Each digital state set is a user-defined list of strings, enumerated from 0 to n to represent different values of discrete data. For more information about PI digital tags and editing digital state sets, see the PI Data Archive manuals.

An interface point that contains discrete data can be stored in the PI Data Archive as a digital point. A digital point associates discrete data with a digital state set, as specified by the user.

Interface Specific Digital Sets

If quality tags are being used, quality point specific digital sets must be created. Example structure files are provided to assist in the configuration. Configuration files for the following digital state sets are provided.

- PIC37118_CompositeQual
- PIC37118_ConfigChange
- PIC37118_ConnectState
- PIC37118_DataSorting
- PIC37118_DataVaild
- PIC37118_LeapSecond
- PIC37118_NominalFreq
- PIC37118_PMUError
- PIC37118_SyncError
- PIC37118_Timelock
- PIC37118_TimeQuality
- PIC37118_Trigger

The example structure files provided can be used to import the sets into PI using SMT. They are located in the `Configuration Files` subdirectory of the specified installation location.

System Digital State Set

Similar to digital state sets is the system digital state set. This set is used for all points, regardless of type, to indicate the state of a point at a particular time. For example, if the interface receives bad data from the data source, it writes the system digital state `Bad Input` to the PI point instead of a value. The system digital state set has many unused states that can be used by the interface and other PI clients. Digital States 193-320 are reserved for OSIsoft applications.

Chapter 6. PointSource

The PointSource is a unique, single or multi-character string that is used to identify the PI point as a point that belongs to a particular interface. For example, the string *Boiler1* may be used to identify points that belong to the *MyInt* interface. To implement this, the PointSource attribute would be set to Boiler1 for every PI point that is configured for the *MyInt* interface. Then, if **/ps=Boiler1** is used on the startup command-line of the *MyInt* interface, the interface will search the PI Point Database upon startup for every PI point that is configured with a PointSource of Boiler1. Before an interface loads a point, the interface usually performs further checks by examining additional PI point attributes to determine whether a particular point is valid for the interface. For additional information, see the **/ps** parameter. If the PI API version being used is earlier than 1.6.x or the PI Data Archive version is earlier than 3.4.370.x, the PointSource is limited to a single character unless the SDK is being used.

Case-sensitivity for PointSource Attribute

The PointSource character that is supplied with the **`/ps`** command-line parameter is not case sensitive. That is, **`/ps=P`** and **`/ps=p`** are equivalent.

Reserved Point Sources

Several subsystems and applications that ship with the PI System are associated with default PointSource characters. The Totalizer Subsystem uses the PointSource character `T`, the Alarm Subsystem uses @ for Alarm points, `G` for Group Alarms and `Q` for SQC Alarm points, Random uses `R`, RampSoak uses `9`, and the Performance Equations Subsystem uses `C`. Do not use these PointSource characters or change the default point source characters for these applications. Also, if a PointSource character is not explicitly defined when creating a PI point; the point is assigned a default PointSource character of `Lab` (PI 3). Therefore, it would be confusing to use `Lab` as the PointSource character for an interface.

Note: Do not use a point source character that is already associated with another interface program. However it is acceptable to use the same point source for multiple instances of an interface.

Chapter 7. PI Point Configuration

The PI point is the basic building block for controlling data flow to and from the PI Data Archive. A single point is configured for each measurement value that needs to be archived.

Point Attributes

Use the point attributes below to define the PI point configuration for the interface, including specifically what data to transfer.

This document does not discuss the attributes that configure UniInt or PI Data Archive processing for a PI point. Specifically, UniInt provides exception reporting and the PI Data Archive or PIBufss provides data compression. Exception reporting and compression are very important aspects of data collection and archiving, which are not discussed in this document.

Note: See the *PI Universal Interface (UniInt) User Guide* and PI Data Archive documentation for information on other attributes that are significant to PI point data collection and archiving.

Tag

The Tag attribute (or tag name) is the name for a point. There is a one-to-one correspondence between the name of a point and the point itself. Because of this relationship, PI System documentation uses the terms “tag” and “point” interchangeably.

Follow these rules for naming PI points:

- The name must be unique in the PI Data Archive.
- The first character must be alphanumeric, the underscore (_) or the percent sign (%).
- Control characters such as linefeeds or tabs are illegal.
- The following characters also are illegal: * ’ ? ; { } [ ] | \ ` ‘ “

Length

Depending on the version of the PI API and the PI Data Archive, this interface supports Tag attributes whose length is at most 255 or 1023 characters. The following table indicates the maximum length of this attribute for all the different combinations of PI API and PI Data Archive versions.

PI API	PI Data Archive	Maximum Length
1.6.0.2 or later	3.4.370.x or later	1023
1.6.0.2 or later	Earlier than 3.4.370.x	255
Earlier than 1.6.0.2	3.4.370.x or later	255
Earlier than 1.6.0.2	Earlier than 3.4.370.x	255

If the PI Data Archive version is earlier than 3.4.370.x or the PI API version is earlier than 1.6.0.2, and you want to use a maximum Tag attribute length of 1023, you need to enable the PI SDK. See Appendix B. PI SDK Options for information.

PointSource

The PointSource attribute contains a unique, single or multi-character string that is used to identify the PI point as a point that belongs to a particular interface. For additional information, see the **`/ps`** command-line parameter and the PointSource chapter.

PointType

Typically, device point types do not need to correspond to PI point types. For example, integer values from a device can be sent to floating point or digital PI tags. Similarly, a floating-point value from the device can be sent to integer or digital PI tags, although the values will be truncated.

Float16, float32, float 64, int16, int32, digital, string, and blob point types are supported. For more information on the individual point types, see the PI Data Archive manuals.

Location1

Location1 indicates to which copy of the interface the point belongs. The value of this attribute must match the **`/id`** command-line parameter.

Location2

Location2 specifies the ID of the LocalEndPoint. This corresponds to the `ID` parameter of the corresponding LocalEndPoint section in the interfaces XML configuration file. This attribute is used by the interface to uniquely identify the LocalEndPoint object and does not correspond to any C37.118 attributes.

Note: The ID Must Be Unique among all LocalEndPoint sections in the configuration file.

Location3

Location3 specifies the ID of the RemoteEndPoint. This corresponds to the IdCode attribute of the corresponding RemoteEndPoint section in the interfaces XML configuration file and must match the configured IDCODE on the PDC / PMU hardware. If the C37.118 device is a PDC there will most likely be multiple PMUs. This attribute should match the IDCODE field for the PDC and not the PMU.

Location4

The C37.118 interface is exception based only. Location 4 should be set to zero for all points.

Location5

Location5 specifies the PMU ID. This attribute must match the IDCODE field for the PMU in the C37.118 CONFIG2 message as well as the IdCode attribute in the PMU section of the XML configuration file. The value for this will be determined when configuring PDC or PMU. If the C37.118 device is a PMU, versus a PDC, this value will typically match the IDCODE for the RemoteEndPoint specified in location3.

If the device is a PDC, each PMU being collected by the PDC must have a unique IDCODE field configured.

InstrumentTag

The InstrumentTag attribute specifies the specific C37.118 measurement to be stored in the PI Data Archive. The attribute contains two elements. The first is the Measurement Type and the second is the specific channel name, REP or PMU measurement and is in the format of **MeasurementType\Measurement**.

Length

Depending on the version of the PI API and the PI Data Archive, this interface supports an `InstrumentTag` attribute whose length is at most 32 or 1023 characters. The following table indicates the maximum length of this attribute for all the different combinations of PI API and PI Data Archive versions.

PI API	PI Data Archive	Maximum Length
1.6.0.2 or later	3.4.370.x or later	1023
1.6.0.2 or later	Earlier than 3.4.370.x	32
Earlier than 1.6.0.2	3.4.370.x or later	32
Earlier than 1.6.0.2	Earlier than 3.4.370.x	32

If the PI Data Archive version is earlier than 3.4.370.x or the PI API version is earlier than 1.6.0.2, and you want to use a maximum InstrumentTag length of 1023, you need to enable the PI SDK. See Appendix B. PI SDK Options for information.

Measurement Types

There are five primary Measurement Types that the interface supports. They are:

- Phasor – The interface will read one of the C37.118 Phasor channels. There are four subtypes for Phasor measurements.
- Analog – The interface will read one of the C37.118 Analog channels
- Digital – The interface will read one of the C37.118 Digital channels.
- PMU – PMU measurement types are those which are not Phasor, Analog or Digital and are related to the PMU. These include PMU values such Frequency (FREQ) and Frequency rate of Change (DFREQ), Configuration data such as the data scheduling time base (TIME_BASE) field from the C37.118 CONFIG message and interface configuration data from the XML configuration file such as the PMU Latitude and Longitude.
- REP – REP measurement types are those related to the RemoteEndPoint. These include connection status as well as time quality.

For Phasor measurement types, the interface can return four different measurements: Real, Imaginary, Magnitude and Phase Angle.

The following table illustrates the valid MeasurementType keywords for use in the InstrumentTag attribute. The keywords are not case sensitive.

Keyword	Description
PhasorReal	Returns the Real measurement for the specified Phasor Channel.
PhasorImag	Returns the Imaginary measurement for the specified Phasor Channel.
PhasorMag	Returns the Magnitude measurement for the specified Phasor Channel.
PhasorAngle	Returns the Phase Angle measurement for the specified Phasor Channel
Analog	Returns the measurement from the specified Analog Channel.
Digital	Returns the Digital measurement from the specified Digital Channel.
PMU	The interface returns one of the PMU measurements.
REP	Remote End Point related tag

Measurements

For Phasor, Analog and Digital Measurement types, the user must specify the C37.118 channel name from which the interface will collect data. Channel names vary by vendor.

For the PMU and REP Measurement types, the user must enter one of the PMU or REP Measurement keyword described below.

The C37.118 specification defines the channel names to be a 16-byte ASCII string that can have embedded as well as trailing spaces. The interface strips any trailing spaces but leaves any embedded spaces. The interface also treats the strings as case insensitive. If for example a PMU has a channel name of ‘Watts CH-A ‘, the user would specify ‘Watts CH-A’ for the Measurement element of the InstrumentTag (‘watts ch-a’ would also be valid in this example).

For PMU MeasurementTypes tags, the interface will store various data from the C37.118 data stream, CONFIG1 and CONFIG2 blocks and configuration data from the XML configuration file. The follow table shows the valid values for the Measurement field when the MeasurmentType field is PMU.

Keyword	Description
FREQ	System frequency (FREQ) from C37.118 data stream
DFREQ	Frequency Rate of Change (DFREQ) from C37.118 data stream.
FREQERR	Frequency error calculated from FREQ – Nominal Frequency as defined by FNOM (50 or 60).
CFGCNT	Configuration change count (CFGCNT) from last CONFIG block
CONFIGFLAG	Digital tag that is set to 1 when the configuration change flag (bit 10) in the STAT word is true.
FNOM	Nominal line frequency (FNOM) from last CONFIG block
DATA_RATE	Date rate (DATA_RATE) from last CONFIG block
LATITUDE	PMU Latitude (-90.0-90.0) from XML configuration file
LONGITUDE	PMU Longitude (-180.0-180.0) from XML configuration file
TIMELOCK	Digital tag receives the Unlocked Time quality bits (5-4) from the STAT word. A sample digital set definition file **`PIC37118_Timelock.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
TRIGGER	Digital tag receives the Trigger Reason bits (3-0) from the STAT word. A sample digital set definition file **`PIC37118_Trigger.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
DATAVALID	Digital tag receives the Data Validity bit (15) from the STAT word. A sample digital set definition file **`PIC37118_DataValid.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
PMUERROR	Digital tag receives the PMU Error bit (14) from the STAT word. A sample digital set definition file **`PIC37118_PMUError.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
SYNCERROR	Digital tag receives the PMU Sync Error bit (13) from the STAT word. A sample digital set definition file **`PIC37118_SyncError.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
DATASORTING	Digital tag receives the Data Sorting bit (12) from the STAT word. A sample digital set definition file **`PIC37118_DataSorting.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
COMPOSITEQUAL	Digital tag receives a composite status from the Data Valid, PMU Error, Sync Error and Data Sorting flags. If any of these flags are set, the value of this tag will be set to 1. Otherwise it will be zero. This tag is intended to be used to enable the user to make a single check for any quality related issues and then do further checking only if needed. A sample digital set definition file **`PIC37118_CompositeQuality.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.

The following table lists measurement values for PMU measurement type tags.

Keyword	Description
TIMEQUAL	Digital tag receives the Qime Quality bits (3-0) from the time quality flags. A sample digital set definition file **`PIC37118_TimeQuality.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
LEAPSEC	Digital tag receives the Leap Second quality bits (6-4) from the time quality flags. A sample digital set definition file **`PIC37118_LeapSecond.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
CONNECTSTATE	Digital tag receives the current connection state. A sample digital set definition file **`PIC37118_ConnectState.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
EVENTSEQ	Integer tag intended for diagnostic purposes to verify that the interface is not losing messages. It records the sequence number of the message within a second. For example: If the update rate for the PMU is set to 30hz, the expected values would cycle from 0 to 29 without missing any values.

ExDesc

This attribute is not used by the PI C37.118 interface.

Scan

By default, the Scan attribute has a value of 1, which means that scanning is turned on for the point. Setting the scan attribute to 0 turns scanning off. If the scan attribute is 0 when the interface starts, a message is written to the log and the point is not loaded by the interface. There is one exception to the previous statement:

If any PI point is removed from the interface while the interface is running (including setting the scan attribute to 0), `SCAN OFF` will be written to the PI point regardless of the value of the Scan attribute. Two examples of actions that would remove a PI point from an interface are to change the point source or set the scan attribute to 0. If an interface-specific attribute is changed that causes the point to be rejected by the interface, `SCAN OFF` will be written to the PI point.

Shutdown

The Shutdown attribute is 1 (true) by default. The default behavior of the PI Shutdown subsystem is to write the `SHUTDOWN` digital state to all PI points when PI is started. The timestamp that is used for the `SHUTDOWN` events is retrieved from a file that is updated by the Snapshot Subsystem. The timestamp is usually updated every 15 minutes, which means that the timestamp for the `SHUTDOWN` events will be accurate to within 15 minutes in the event of a power failure. For additional information on shutdown events, refer to PI Data Archive manuals.

Note: The `SHUTDOWN` events that are written by the PI Shutdown subsystem are independent of the `SHUTDOWN` events that are written by the interface when the **`/stopstat=Shutdown`** command-line parameter is specified.

`SHUTDOWN` events can be disabled from being written to PI points when the PI Data Archive is restarted by setting the Shutdown attribute to 0 for each point. Alternatively, the default behavior of the PI Shutdown Subsystem can be changed to write `SHUTDOWN` events only for PI points that have their Shutdown attribute set to 0. To change the default behavior, edit the `\PI\dat\Shutdown.dat` file, as discussed in PI Data Archive manuals.

Bufserv and PIBufss

It is not recommended to write shutdown events when buffering is being used. Bufserv and PIBufss are utility programs that provide the capability to store and forward events to a PI Data Archive, allowing continuous data collection when the PI Data Archive is down for maintenance, upgrades, backups, or unexpected failures. That is, when the PI Data Archive is shut down, Bufserv or PIBufss will continue to collect data for the interface, making it undesirable to write `SHUTDOWN` events to the PI points for this interface. Disabling Shutdown is recommended when sending data to a Highly Available PI Data Archive Collective. Refer to the Bufserv or PIBufss manuals for additional information.

DataSecurity

The PI identity in the PI trust that authenticates the interface must be granted read access by the DataSecurity attribute of every PI point that the interface services. If the interface is used *without* a buffering application, write access also must be granted. (If the interface is used with a buffering application, the buffering application requires write access but the interface does not.)

PtSecurity

The PI identity in the PI trust that authenticates the interface must be granted read access by the PtSecurity attribute of every PI point that the interface services.

Chapter 8. Startup Command File

Command-line parameters can begin with a `/` or with a `-`. For example, the **`/ps=M`** and **`-ps=M`** command-line parameters are equivalent.

Command file names have a .bat extension. The Windows continuation character (`^`) allows for the use of multiple lines for the startup command. The maximum length of each line is 1024 characters (1 kilobyte). The number of parameters is unlimited, and the maximum length of each parameter is 1024 characters.

The PI Interface Configuration Utility (PI ICU) provides a tool for configuring the interface startup command file.

Configuring the Interface with PI ICU

Note: PI ICU requires PI 3.3 or later.

The PI Interface Configuration Utility provides a graphical user interface for configuring PI interfaces. If the interface is configured by the PI ICU, the batch file of the interface (`PIC37118.bat`) will be maintained by the PI ICU and all configuration changes will be kept in that file and the module database. The procedure below describes the necessary steps for using PI ICU to configure the C37118 interface.

From the PI ICU menu, select *Interface*, then *NewWindows Interface Instance from EXE...*, and then *Browse* to the `PIC37118.exe` executable file. Then, enter values for *Host PI System*, *Point Source* and *Interface ID#*. A window such as the following results:

Interface name as displayed in the ICU (optional) will have PI- pre-pended to this name and it will be the display name in the services menu.

Click *Add*.

The following message displays:

Note that in this example the host PI Data server is JMURASKI7440. To configure the interface to communicate with a remote PI Data server, select *Connections…* on the PI ICU *Interface* menu and select the default server. If the remote node is not present in the list of servers, it can be added.

After the interface is added to PI ICU, near the top of the main PI ICU screen, the Interface *Type* should be `C37118`. If not, use the drop-down box to change the Interface *Type* to be `C37118`."

Click *Apply* to enable the PI ICU to manage this copy of the C37118 interface.

The next step is to make selections in the interface-specific tab (i.e. "`C37118`") that allow the user to enter values for the startup parameters that are particular to the C37118 interface.

Since the C37118 interface is a UniInt-based interface, in some cases the user will need to make appropriate selections in the *UniInt* page. This page allows the user to access UniInt features through the PI ICU and to make changes to the behavior of the interface.

To set up the interface as a Windows Service, use the *Service* page. This page allows configuration of the interface to run as a service as well as to starting and stopping of the interface service. The interface can also be run interactively from the PI ICU. To do that, select *Start Interactive* on the *Interface* menu.

For more detailed information on how to use the above-mentioned and other PI ICU pages and selections, please refer to the *PI Interface Configuration Utility* user guide. The next section describes the selections that are available from the *C37118* page. Once selections have been made on the PI ICU GUI, press the *Apply* button in order for PI ICU to make these changes to the interface's startup file.

C37118 Interface page

Since the startup file of the C37118 interface is maintained automatically by the PI ICU, use the *C37118* page to configure the startup parameters; you should avoid making changes to the file manually. The following is the description of interface configuration parameters that are used in the PI ICU Control and the corresponding manual parameters.

C37118

The PI Interface for IEEE C37.118 - ICU has one tab. A yellow text box indicates that an invalid value has been entered, or that a required value has not been entered.

Path to XML Config File

This box is used to specify the path and filename of the XML Configuration file to be read by the interface. If the path contains spaces, then the path and filename should be enclosed in quotation marks. To enter a <UNC Path> click the Browse button and choose an existing file from the resulting dialog box (shown below) or enter a new filename in the File name: text box at the bottom of the screen **(/CONFIGFILE=***<UNC Path>***)**

Additional Parameters

This section is provided for any additional parameters that the current ICU Control does not support.

XML Parameters

Session

The session section contains configuration parameters that control the behavior of the interface when various error conditions occur.

The following figure illustrates the Session section of the C37.118 ICU.

- Sync Error Action – Controls the behavior of the interface when the SYNC flag is set in the C37.118 data stream. (`SyncErrorAction=#`, Default: 1 (Mark as Questionable), Range 0-3)
- Data Sorting Action – Controls the behavior of the interface when the Data Sorting flag is set in the C37.118 data stream. This flag is set on in a PDC data stream when a PMU loses time synchronization making it impossible for the PDC to time correlate the Phasor data. (`DataSortingAction=#`, Default: 1 (Mark as Questionable), Range 0-3)
- Data Valid Action – Controls the behavior of the interface when the Data Valid flag is set in the C37.118 data stream. This flag is set in a PDC data stream when the PDC loses connection to a PMU. When this happens, the PDC typically zero fills the data. (`DataValidAction=#`, Default: 3 (Disard the Data), Range 0-3)
- PMU Error Action – Controls the behavior of the interface when the PMU Error flag is set in the C37.118 data stream. This flag is set when an error occurs on a PMU and indicates a possible error in the data. (`PMUErrorAction=#`, Default: 3 (Discard the Data), Range 0-3)
- Leap Second Action – Controls the behavior of the interface when the Leap Second flag is set in the C37.118 data stream. This flag indicates that a leap second has occurred. (`LeapSecondAction=#`, Default: 0 (Store the Value), Range 0-3)
- Write IO Timeout – If set to yes, the interface will write the system digital state of I/O Timeout when no C37.118 data messages are received within the timeframe specified in the LocalEndPoint section. (`WriteIOTimeout=#`, Default: 1 (Yes), Range:0-1)
- Skip CONFIG2 Command – If set to yes, the interface will send CONFIG2 command to PDC /PMU only during startup. Otherwise, the interface will send CONFIG2 request each time when it is trying to reconnect to PDC / PMU. (`SkipConfig2Cmd=#`, Default: 0 (No), Range:0-1)
- Flush Events – If set to yes, the interface will flush all data to the PI Data Archive after each C37.118 data messages received from PDC /PMU. Otherwise, it will flush

data when buffer gets full (192 records) (`FlushEvents=#`, Default: 0 (No), Range:0-1).

- Send Stop Data Command – If set to yes, the interface will send DATA_STOPCOMMAND to the PDC / PMU when user shutdown the interface (`SendStopCmd=#`, Default: 0 (NO), Range:0-1)
- Switch To Backup Delay – Controls the behavior of the interface for the PDC /PMU Data Source Level Failover. If the interface is not able to set connection and receive data from primary PDC / PMU during this period of time it will switch and try to connect to the backup PDC / PMU. If it loses connection to backup and is not able to reconnect to backup PDC / PMU again during this period of time it will try to reconnect to the primary PDC / PMU again. (`SwitchToBackupDelay=#`, Default: 60 sec (Store the Value), Range 30-600)

The available selections for Sync Error, Data Sorting, Data Valid, PMU Error and Leap Second Actions above are as follows.

- 0 (Store the Value) – The interface will store the value normally.
- 1 (Mark as Questionable) – The interface will store the value but mark it as questionable.
- 2 (Store the Corresponding System Digital State) – The interface will store the corresponding system state.
- 3 (Discard the Data) – The interface will not store the value in the PI Data Archive.

LocalEndPoint

The LocalEndPoint section defines parameters specific to the interface side of a PDC / PMU connection.

The following illustrates the format of the LocalEndPoint section of the C37.118 ICU.

- IP – Identifies this LocalEndPoint as using a TCP connection (`Type=<string>`, Range: IP or Serial)
- Serial – Identifies this LocalEndPoint as using a Serial connection (`Type=<string>`, Range: IP or Serial)
- ID – Specifies the numeric ID of the LocalEndPoint. The ID uniquely identifies the LocalEndPoint when configuring tags for an interface. The ID must be unique among all LocalEndPoints. (`Id=#`, Default:None, Range: 0-65535, Required)
- Comm Dll – Defines the Communications Plug-In to be used to communicate with the PDC / PMU. Due to the variation in application level functionality from one device vendor to another, the plug-in approach allows for new devices to be easily supported. Refer to Appendix C. Communication Plug-In Selection in this manual or to the OSIsoft support site for details on available plug-in's.
- Data Timeout – Controls the length of time to wait, in milliseconds, for data responses. The recommended value is 5 times the configured PMU data rate. (`DataTimeout=#`, Default:1000, Range: 10-65,535)
- Enabled – Allows the LocalEndPoint and everything associated with it to be disabled and not loaded by the interface. It can then be re-enabled with having to reconfigure. (`Enabled=#`, Default: 1 (Yes), Range: 0-1)
- Reconnect Rate – Controls the rate in which the interface attempts to re-establish communications with the PDC / PMU. (`ReconnectRate=#`, Default:30000, Range: 10-65,535)
- Primary – Allows to specify which LocalEndPoint in the redundant pair becomes a primary (value should be 1). For another LocalEndPoint this field should have opposite value (0) to become a backup. Use this field to setup Data Source Level Failover. (`Primary=#`, Default: 0 (No), Range: 0-1)
- Failover ID – Specifies the numeric Failover ID of the LocalEndPoint. The Failover ID allows creating redundant pair of primary and secondary LocalEndPoints for Data Source Level Failover. For example, if ID for LocalEndPoint is 1and Failover ID is 2 then another for LocalEndPoint with ID 2 and Failover ID 1 has to be created. In this case LocalEndPoint 1 and 2 becomes redundant pair. You should specify which

LocalEndPoint is primary using Primary field. Only one LocalEndPoint in the redundant pair could be a primary. Another one becomes a secondary. If no need to setup Data Source Level Failover for this LocalEndPoint leave this field blank. Use this field only in case to setup Data Source Level Failover. (`FailOverId=#`, Default:None, Range: 0-65535, Optional)

IP Configuration

The following controls are available when Type="IP" is selected.

- Command Protocol – Specifies the protocol to use for C37.118 Command and Configuration messages. (`CommandProtocol=<type>`, Default: TCP, Range: TCP or UDP)
- Local Command Port – Specifies the local port to listen on for configuration packets. If not specified the interface will not specify a specific port (system assigned). This parameter will typically be used when the command protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the Local Data Port value. (`LocalCommandPort=#`, Default: 0 (System Assigned), Range: 1024-65,535)
- Data Protocol – Specifies the protocol to use for C37.118 data messages. (`DataProtocol=<type>`, Default: UDP, Range: TCP or UDP)
- Local Data Port – Specifies the local port to listen on for data packets. If not specified the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. For some PDC / PMU configurations in which the device is sends to a predefined data port, this value will also need to be specified. It is also required if Multicast being used. (`LocalDataPort=#`, Default: 0 (System Assigned), Range: 1024-65,535)
- Close Command Socket – If Yes is specified, the interface will close the command socket (if different than the data socket) once a command is sent and the expected response is successfully received. Some devices, such as the Arbiter 1133, expect that the command socket be closed after use. Others, such as SEL devices require the socket remain open. (`CloseCommandSocket=#`, Default: 1 (Yes), Range: 0 (No) – 1 (Yes))
- Use Local IP Address – This parameter could be used if interface node has multiple IP addresses. When specified, the interface will be binding with that address. Otherwise the interface will be using default IP address. (`LocalIPAddress=<ipaddr>`, Default: None)
- Use Multicast IP Addr – This parameter is valid only when the Local Data Protocol is specified as UDP. When specified, the interface will be configured to listen for C37.118 data messages via Multicast. (`MulticastIPAddress=<ipaddr>`, Default: None)

- Src Specific Multicast – This parameter is valid only if the Local IP Address and Multicast IP Address fields have been populated. When specified, the interface will subscribe for multicast data from the source that has specified IP address. (**`MulticastSourceIPAddress=`**`<ipaddr>`, Default: None)

Serial Configuration

The following controls are available when Type="Serial" is selected.

- Baud – Specifies speed at which to communicate with the PDC / PMU. Refer to your PDC / PMU hardware users guide for supported baud rates. (`Baud=#`, Default: 38400, Range: 300-115200)
- COM Port – The numeric value of the serial COM port to use. For example if using COM1 set this value to 1. (`Port=#`, Range: 1-Max Ports)
- Parity – Specifies the error checking parity to use. (`Parity=#`, Default: 0 (NONE), Values: 0 (NONE), 1 (ODD), 2 (EVEN), 3 (MARK), and 4 (SPACE))
- Stop Bits – Specifies the number of stop bits following a message. (`StopBits=#`, Default: 0 (1 bit), Values: 0 (1 bit), 1 (2 bits), 2 (1.5 bits))

RemoteEndPoint

The RemoteEndPoint represents the remote (PDC / PMU side) of a physical connection. Each RemoteEndPoint node will have one or more PMU child nodes. Each of the PMU nodes will represent a PMU that the PI C37.118 interface will communicate with through the physical port described by the LocalEndPoint / RemoteEndPoint child node pair.

RemoteEndPoint Child Nodes that apply to all connection types.

- ID Code – Specifies the numeric ID of the PDC or PMU. This must match the IDCODE field sent by the C37.118 device. If the device is a PDC it is important to note that this must match the IDCODE of the PDC and not the PMU(s) being serviced by the PDC. (`IdCode=#`, Range: 0-32,767)
- Description – User defined description of the RemoteEndPoint. Ie. "Substation xyz". This attribute is used by the interface when logging messages regarding the RemoteEndPoint. (`Description=`*`<string>`*)

The following illustrates the RemoteEndPoint section for an IP LocalEndPoint.

- Remote Data Port – Specifies the remote (PDC) port to be used for data packets. If not specified the interface will assume a specific port (system assigned). If this field is specified, the data being received from the PDC / PMU must originate from this port. Most PDCs and PMUs use a random system assigned port. (`RemoteDataPort=#`, Default: 4712, Range: 1024-65,535)
- Remote Command Port – Specifies the remote port to use for command and configuration packets. The interface will send CONFIG1, CONFIG2 and HEADER request commands to this port. (`RemoteCommandPort=#`, Default: 4713, Range: 1024-65,535)
- Host Name – If this control is selected the PDC / PMU addressing information is passed by host name.(`HostName=<nodename>` or `<IPAddr>`)
- IP Address – If this control is selected the PDC / PMU addressing information is passed as an IP address in dotted decimal notation. (`HostName=<nodename>` or `<IPAddr>`)

The following illustrates the RemoteEndPoint section for a Serial LocalEndPoint.

PMU

The PMU section describes the characteristics of each PMU. If the RemoteEndPoint is a PDC then there may be multiple PMUs, otherwise there should be only one PMU child node per RemoteEndPoint.

- ID Code – Specifies the numeric ID of the PMU. This must match the IDCODE field sent by the C37.118 device. If there are multiple PMU nodes associated with a RemoteEndPoint (RemoteEndPoint is a PDC) with it this ID must be unique within the RemoteEndPoint collection and should match the PMUs IDCODE and not the IDCODE of the PDC itself. (`IdCode=#`, Range: 0-32,767)
- Latitude – Specifies the geographic latitude of the PMU. This field is optional. However, if the latitude is specified, then the longitude must also be specified, otherwise the interface will log an error and exit. If no sign is specified then + is assumed. (`Latitude=+/-#.#`, Range: -90.0 to +90.0)
- Longitude – Specifies the geographic longitude of the PMU. This field is optional. However, if the longitude is specified, then the latitude must also be specified, otherwise the interface will log an error and exit. If no sign is specified then + is assumed. (`Longitude=+/-#.#`, Range: -180.0 to +180.0)

PIC37118 XML Nodes, Elements, and Attributes e

Since the startup file of the C37118 interface is maintained automatically by the PI ICU, use the C37118 page to configure the startup parameters and do not make changes in the file manually. The following is the description of interface configuration parameters used in the PI ICU Control and corresponding manual parameters.

Root Node

The root node must be named PI_C37_118.Session and contains one or more LocalEndPoint leaf nodes.

PI_C37118.Session Child Nodes	Description
Example: <PI_C37_118.Session”>	
WriteIOTimeout Optional XMLType: Attribute Default: 1	By default, the interface will write the digital state I/O Timeout to all measurement type tags when a communications timeout condition is encountered. Setting the WriteIOTimeout attribute to a value of 0 will cause the interface to suppress writing I/O Timeout to the measurement tags. Valid values for this attribute are 0 (No) to not write I/O Timeout or 1 (Yes) to write I/O Timeout.
PMUErrorAction Optional XMLType: Attribute Default: 3	The PMUErrorAction attribute controls the behavior of the interface when the PMU Error flag is detected in the STAT word of the message. This flag indicates that there is a problem with the PMU hardware and that the data may not be valid. The default behavior of the interface is to discard the data. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
DataValidAction Optional XMLType: Attribute Default: 3	The DataValidAction attribute controls the behavior of the interface when the Data Invalid flag is detected in the STAT word of the message. This flag indicates that there is a problem with the data for the PMU may not be valid. This flag is typically set when the data stream is coming from a PDC versus a single PMU and is usually due to the PDC not receiving a data frame from the PMU in a reasonable time. The PDC typically blank fills the data. The default behavior of the interface is to discard the data. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values 3 – Discards the data.
SyncErrorAction Optional XMLType: Attribute Default: 1	The SyncErrorAction attribute controls the behavior of the interface when the PMU Sync Error flag is detected in the STAT word of the message. This flag indicates that there is a problem with the time synchronization of the PMU and that the quality of the timestamp is other than normal. The default behavior of the interface is to mark the data as questionable. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.

PI_C37118.Session Child Nodes	Description
Example: <PI_C37_118.Session”>	
DataSortingAction Optional XMLType: Attribute Default: 1	The DataSortingAction attribute controls the behavior of the interface when the Data Sorting flag is detected in the STAT word of the message. This error should only occur if the data stream is coming from a PDC. This flag indicates that the data for the PMU is being integrated into the data frame by arrival and not timestamp. The default behavior of the interface is to mark the data as questionable. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
LeapSecondAction Optional XMLType: Attribute Default: 0	The LeapSecondAction attribute controls the behavior of the interface when one of the leap second flags is detected in the STAT word of the message. This occurs when a leapsecond is occurring. The default behavior of the interface is to store the data normally. Valid selections for this attribute are: 0 – Store the value normally 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
SkipConfig2Cmd XML Type: Attribute Default: 0	The SkipConfig2Cmd attribute controls the behavior of interface to send CONFIG2 command to PDC / PMU each time when it is trying to reconnect to PDC / PMU. If set to yes, the interface will send CONFIG2 command to PDC / PMU only during startup. Valid values for this attribute are 0 (No) to not skip sending CONFIG2 request or 1 (Yes) to skip sending CONFIG2 request.
SendStopCmd XML Type: Attribute Default: 0	The SendStopCmd attribute controls the behavior of interface to send STOP_DATA command to the PDC / PMU when user shutdown the interface. It may affect other clients that connected to that PMU / PDC. Valid values for this attribute are 0 (No) to not send STOP_DATA command or 1 (Yes) to send STOP_DATA command when shutting down the interface.
FlushEvents XML Type: Attribute Default: 0	The FlushEvents attribute controls the behavior of interface to flush all data to the PI Data Archive after each C37.118 data messages received from PDC / PMU. If set to 0 (default value) it will flush data when buffer gets full (192 records). Valid values for this attribute are 0 (No) to not flush data or 1 (Yes) to flush all data to PI Data Archive after each data message received from PMU / PDC.
SwitchToBackupDelay XML Type: Attribute Default: 60	The SwitchToBackupDelay controls the behavior of the interface for the PDC / PMU Data Source Level Failover. If the interface is not able to set connection and receive data from primary PDC / PMU during this period of time it will switch and try to connect to the backup PDC / PMU. If it loses connection to backup and is not able to reconnect to backup PDC / PMU again during this period of time it will try to reconnect to the primary PDC / PMU again. Valid ranges: {30-600}

LocalEndPoint Nodes

Child nodes of the root node are LocalEndPoint nodes. The LocalEndPoint nodes are used to specify the properties of the local (interface side) of a physical connection such as a communication port, or TCP port. The **Type** attribute must be either `Serial` or **`IP`**. The child nodes will depend on the value of the **Type** attribute. A serial communications port will have different child nodes from an IP port. There can only be one RemoteEndPoint Node for each LocalEndPoint Node.

LocalEndPoint Child Nodes that apply to all connection types.	**Description**
Example: <RemoteEndPoint>	
Type Required XML Type: Element Default: None	Specifies the communications type. Values are “IP” or “Serial”
Id Required XMLType: Attribute Default: None	Specifies the numeric ID of the LocalEndPoint. The ID uniquely identifies the LocalEndPoint when configuring tags for an interface. The ID must be unique among all LocalEndPoints for an interface. Valid ranges: {0-65535 }
Enabled Optional XML Type: Attribute Default: 1	Enabled – Allows the LocalEndPoint and everything associated with it to be disabled and not loaded by the interface. It can then be re-enabled with having to reconfigure. A value of 0 (No) indicates that the LocalEndPoint is disabled. A value of 1 (Yes) indicates it is enabled.
CommDll Required XMLType: Element Default: None	Defines the Communications Plug-In to be used to communicate with the PDC / PMU. Due to the variation in application level functionality from one device vendor to another, the plug-in approach allows for new devices to be easily supported. Refer to Appendix C. Communication Plug-In Selection in this manual or to the OSIsoft support site for details on available plug-in’s.
DataTimeout Optional XMLType: Attribute Default: 1000 (1 second)	Controls the length of time to wait, in milliseconds, for data responses. Valid range: { 10-65535}
ReconnectRate Optional XMLType: Attribute Default: 30000 (30 seconds)	Controls the interval, in milliseconds, to be used when attempting to restart communications with a C37.118 device. Valid Range: {10-65535}
FailOverId Optional XML Type: Attribute Default: None	Specifies the numeric Failover ID of the LocalEndPoint. The Failover ID allows creating redundant pair of primary and secondary LocalEndPoints for Data Source Level Failover. Valid Range: {0-65535 or empty}
Primary Optional XML Type: Attribute Default: None	Controls which LocalEndPoint in the redundant pair becomes a primary (value should be 1). For another LocalEndPoint this field should have opposite value (0) to become a backup. Use this field to setup Data Source Level Failover. This field is disabled on ICU control if FailOverID is empty. Valid Range {0-1 or disabled}

LocalEndPoint Child Nodes that apply to all connection types.	Description
Example: <RemoteEndPoint>	
Type Required XML Type: Element Default: None	Specifies the communications type. Values are “IP” or “Serial”
LocalEndPoint Child Nodes for Type=”Serial”	**Description**
Example: <LocalEndPoint Type=”Serial”>	
Port Required XMLType: Attribute Default: None	The numeric value of the serial COM port to use. For example if using COM1 set this value to 1. Valid range : {1 – Max Ports}
Baud Optional XMLType: Attribute Default: 38400	The speed at which to communicate. Valid range: {Refer to your PDC / PMU hardware users guide for supported baud rates.}
Parity Optional XMLType: Attribute Default: None	The error checking parity to use. This can be 0 for NONE, 1 for ODD, 2 for EVEN, 3 for MARK, and 4 for SPACE. Valid range: {0,1,2,3,4}
StopBits Optional XMLType: Attribute Default: None	The number of stop bits following a message. This can be 0 for one, 1 for two, or 2 for one and a half stop bits. Valid range: {0,1,2}

LocalEndPoint Child Nodes for Type=”IP”	Description
Example: <LocalEndPoint Type=”IP”>	
DataProtocol Optional XMLType: Element Default: UDP	Specifies the protocol to use for C37.118 data messages. Valid Values: {UDP or TCP}
LocalDataPort Required if Multicast being used, otherwise optional XMLType: Attribute Default: 0 (System Assigned)	The local port to bind to for data packets. If not specified or zero the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the LocalCommandPort value or zero. Valid range: {1024-65535}
MulticastIPAddress Optional XMLType: Element Default: None	This parameter is valid only when the DataProtocol is specified as UDP. When specified, the interface will be configured to listen for C37.118 data messages via Multicast. Valid values: {224.0.0.1-224.255.255.254}

LocalEndPoint Child Nodes for Type="IP"	Description
Example: <LocalEndPoint Type="IP">	
CommandProtocol Optional XMLType: Element Default: TCP	Specifies the protocol to use for C37.118 Command and Configuration messages. If this node is not specified, the interface will use the protocol specified by the DataProtocol node or the default DataProtocol if it is not specified. Valid Values: {UDP or TCP}
LocalCommandPort Optional XMLType: Attribute Default: 0 (System Assigned)	The local port to bind to for configuration packets. If not specified or zero the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the LocalDataPort value or zero. Valid range: {1024-65535}
CloseCommandSocket Optional XML Type: Attribute Default Value: 1	Some PMU / PDC models require the socket used for sending commands and receiving configuration and header responses be closed after use, while others require that they remain open. The Arbiter 1133 for example, requires that the socket be closed. All SEL PMU and PDC models require the socket remain open. This option only effects configurations in which the command/command response is on a separate protocol and/or port from the C37.118 data stream. Valid Values: 0 (Leave Open) or 1 (Close After Use)
LocalIPAddress Optional XMLType: Element Default: None	This parameter could be used if interface node has multiple IP addresses. When specified, the interface will bind all ports with that address. Otherwise the interface will be using default IP address. Valid Values: Any IP address that could be used by the interface node or empty.
MulticastSourceIPAddr ess Optional XML Type: Element Default Value: None	This parameter is valid only if the Local IP Address and Multicast IP Address fields have been populated. When specified, the interface will subscribe for multicast data coming from the source that has specified IP address.

RemoteEndPoint Nodes

The RemoteEndPoint node is child node of the LocalEndPoint node(s). The RemoteEndPoint node represents the remote (PDC / PMU side) of a physical connection. The RemoteEndPoint node will have one or more PMU child nodes. Each of the PMU nodes will represent a PMU that the PI C37.118 interface will communicate with through the physical port described by the LocalEndPoint / RemoteEndPoint child node pair.

RemoteEndPoint Child Nodes that apply to all connection types.	Description
Example: <RemoteEndPoint>	
IdCode Required XMLType: Attribute Default: None	Specifies the numeric ID of the PDC or PMU. This must match the IDCODE field sent by the C37.118 device. If a LocalEndPoint has multiple RemoteEndPoint nodes associated with it this ID must be unique within the LocalEndPoint collection. Valid ranges: {0-65535 }

RemoteEndPoint Child Nodes that apply to all connection types.	Description
Description Optional XML Type: Element Default: None	User defined description of the RemoteEndPoint. Ie. “Substation xyz”. This attribute is used by the interface when logging messages regarding the RemoteEndPoint.

RemoteEndPoint Child Nodes when LocalEndPoint is “Serial”	Description
Example: <RemoteEndPoint>	
	There are no Serial specific RemoteEndPoint child nodes

RemoteEndPoint Child Nodes when LocalEndPoint is “IP”	Description
Example: <RemoteEndPoint>	
HostName Required XMLType: Element Default: None	The Host name or IP address of the PDC or PMU. If using the PIC37118_Comm003.DLL, the supplied name should be set to 127.0.0.1 to indicate the interface will listen on the specified LocalDataPort for incoming C37.118 messages. Valid range: {Any valid address or name} Example: <Hostname> 192.164.55.128</Hostname> or <Hostname> myhost </Hostname>
RemoteDataPort Optional XMLType: Attribute Default: 4712	The remote port to be used for data packets. If not specified the interface will not specify a specific port (system assigned). Valid range: {1024-65535} or 0 (System Assigned)
RemoteCommandPort Optional XMLType: Attribute Default: 4713	The remote port to use for command configuration packets. Valid range: {1024-65535}

PMU Child Nodes

PMU nodes are child nodes of the RemoteEndPoint nodes. The PMU node describes the characteristics of the PMU. If the RemoteEndPoint is a PDC then there may be multiple PMUs, otherwise there should be only one PMU child node per RemoteEndPoint.

PMU Child Nodes	Description
Example: <PMU>	
IdCode Required XMLType: Attribute Default: None	Specifies the numeric ID of the PMU. This must match the IDCODE field sent by the C37.118 device. If there are multiple PMU nodes associated with a RemoteEndPoint (RemoteEndPoint is a PDC) with it this ID must be unique within the RemoteEndPoint collection and should match the PMUs IDCODE and not the PDCs. Valid ranges: {0-65535 }
Latitude Optional XMLType: Attribute Default: None	Specifies the geographic latitude of the PMU. If the latitude is specified, then the longitude must also be specified Otherwise the interface will log an error and exit. Valid range: {-90.0 to +90.0}
Longitude Optional XMLType: Attribute Default: None	Specifies the geographic longitude of the PMU. If the latitude is specified, then the longitude must also be specified Otherwise the interface will log an error and exit. Valid range: {-180.0 to +180.0}

Sample PIC37118 XML File

The following is an example file:

```
<?xml version="1.0"?>
<PI_C37_118.Session WriteIOTimeout="1" DataSortingAction="1"
PMUErrorAction="3" SyncErrorAction="1" DataValidAction="3"
LeapSecondAction="0">
   <LocalEndPoint Id="1" LocalDataPort="1024" LocalCommandPort="1025"
DataTimeout="1000">
          <Type>IP</Type>
          <CommDLL>PIC37118_Comm001.dll</CommDLL>
          <CommandProtocol>TCP</CommandProtocol>
          <DataProtocol>UDP</DataProtocol>
          <MulticastIPAddress>239.254.1.2</MulticastIPAddress>
                <RemoteEndPoint IdCode="1">
                      <Description>Arbiter in Savannah</Description>
                      <HostName>gap001</HostName>
                            <PMU IdCode="1" Latitude="37.127652"
Longitude="-122.876365"/>
                            <PMU IdCode="2"/>
                </RemoteEndPoint>
   </LocalEndPoint>
</PI_C37_118.Session>
```

Command-line Parameters

Note: The *PI Universal Interface (UniInt) User Guide* includes details about other command-line parameters, which may be useful.

Parameter	Description
`/CacheMode` Required when using disconnected startup Default: Not Defined	Required for disconnected startup operation. If defined, the **`/CacheMode`** startup parameter indicates that the interface will be configured to utilize the disconnected startup feature.
`/CachePath=`*`path`* Optional Default: Not Defined	Used to specify a directory in which to create the point caching files. The directory specified must already exist on the target machine. By default, the files are created in the same location as the interface executable. If the path contains any spaces, enclose the path in quotes. Examples: **`/CachePath=D:\PIPC\Interfaces\CacheFiles`** **`/CachePath=D:/PIPC/Interfaces/CacheFiles`** **`/CachePath=D:/PIPC/Interfaces/CacheFiles/`** Examples with space in path name: **`/CachePath="D:\Program Files\PIPC\MyFiles"`** **`/CachePath="D:/Program Files/PIPC/MyFiles"`** **`/CachePath="D:/Program Files/PIPC/MyFiles/"`**

/CacheSynch=# Optional Default: 250 ms	**NOTE:** Care must be taken when modifying this parameter. This value must be less than the smallest scan class period defined with the **/f** parameter. If the value of the **/CacheSynch** parameter is greater than the scan class value, input scans will be missed while the point cache file is being synchronized. The optional **/CacheSynch=#** startup parameter specifies the time slice period in milliseconds (ms) allocated by UniInt for synchronizing the interface point cache file with the PI Data Archive. By default, the interface will synchronize the point cache if running in the disconnected startup mode. UniInt allocates a maximum of # ms each pass through the control loop synchronizing the interface point cache until the file is completely synchronized. Synchronization of the point cache file can be disabled by setting the value **/CacheSynch=0**. The minimum synchronization period when cache synchronization is enabled is 50ms Whereas, the maximum synchronization period is 3000ms (3s). Period values of 1 to 49 will be changed by the interface to the minimum of 50ms and values greater than 3000 will be set to the maximum interval value of 3000ms. Default: 250 ms Range: {0, 50 – 3000} time in milliseconds Example: **/CacheSynch=50** (use a 50ms interval) **/CacheSynch=3000** (use a 3s interval) **/CacheSynch=0** (do not synchronize the cache)
/ec=# Optional	The first instance of the **/ec** parameter on the command-line is used to specify a counter number, #, for an I/O Rate point. If the # is not specified, then the default event counter is 1. Also, if the **/ec** parameter is not specified at all, there is still a default event counter of 1 associated with the interface. If there is an I/O Rate point that is associated with an event counter of 1, each copy of the interface that is running without **/ec=#**explicitly defined will write to the same I/O Rate point. This means either explicitly defining an event counter other than 1 for each copy of the interface or not associating any I/O Rate points with event counter 1. Configuration of I/O Rate points is discussed in the section called I/O Rate Point. For interfaces that run on Windows nodes, subsequent instances of the **/ec** parameter may be used by specific interfaces to keep track of various input operations. Subsequent instances of the **/ec** parameter can be of the form **/ec***, where * is any ASCII character sequence. For example, **/ecinput=10** and **/ec=11** are legitimate choices for the second and third event counter strings.
/f=*SS.##* or **/f=***SS.##,ss.##* or **/f=***HH:MM:SS.##* or **/f=***HH:MM:SS.##, hh:mm:ss.##* Required for reading scan-based inputs	The **/f** parameter defines the time period between scans in terms of hours (HH), minutes (MM), seconds (SS) and sub-seconds (##). The scans can be scheduled to occur at discrete moments in time with an optional time offset specified in terms of hours (hh), minutes (mm), seconds (ss) and sub-seconds (##). If HH and MM are omitted, then the time period that is specified is assumed to be in seconds. Each instance of the **/f** parameter on the command-line defines a scan class for the interface. There is no limit to the number of scan classes that can be defined. The first occurrence of the **/f** parameter on the command-line defines the first scan class of the interface; the second occurrence defines the second scan class, and so on. PI points are associated with a particular scan class via the Location4 PI point attribute. For example, all PI points that have Location4 set to 1 will receive input values at the frequency defined by the first scan class. Similarly, all points that have Location4 set

	to 2 will receive input values at the frequency specified by the second scan class, and so on.
	Two scan classes are defined in the following example:
	`/f=00:01:00,00:00:05 /f=00:00:07`
	or, equivalently:
	`/f=60,5 /f=7`
	The first scan class has a scanning frequency of 1 minute with an offset of 5 seconds, and the second scan class has a scanning frequency of 7 seconds. When an offset is specified, the scans occur at discrete moments in time according to the formula:
	scan times = (reference time) + n(frequency) + offset
	where n is an integer and the reference time is midnight on the day that the interface was started. In the above example, frequency is 60 seconds and offset is 5 seconds for the first scan class. This means that if the interface was started at 05:06:06, the first scan would be at **05:07:05**, the second scan would be at **05:08:05**, and so on. Since no offset is specified for the second scan class, the absolute scan times are undefined.
	The definition of a scan class does not guarantee that the associated points will be scanned at the given frequency. If the interface is under a large load, then some scans may occur late or be skipped entirely. See the section “Logging hit, missed, and skipped scans” in the *PI Universal Interface (UniInt) User Guide* for more information on skipped or missed scans.
	`Sub-second Scan Classes`
	Sub-second scan classes can be defined on the command-line, such as
	`/f=0.5 /f=00:00:00.1`
	where the scanning frequency associated with the first scan class is 0.5 seconds and the scanning frequency associated with the second scan class is 0.1 of a second.
	Similarly, sub-second scan classes with sub-second offsets can be defined, such as
	`/f=0.5,0.2 /f=1,0`
	`Wall Clock Scheduling`
	Scan classes that strictly adhere to wall clock scheduling are now possible. This feature is available for interfaces that run on Windows and/or UNIX. Previously, wall clock scheduling was possible, but not across daylight saving time. For example, **`/f=24:00:00,08:00:00`** corresponds to 1 scan a day starting at 8 AM. However, after a Daylight Saving Time change, the scan would occur either at 7 AM or 9 AM, depending upon the direction of the time shift. To schedule a scan once a day at 8 AM (even across daylight saving time), use **`/f=24:00:00,00:08:00,L.`** The `,L` at the end of the scan class tells UniInt to use the new wall clock scheduling algorithm.

/host=*host:port* Required	The **/host** parameter specifies the PI Data Archive node. *Host* is the IP address or the domain name of the PI Data Archive node. *Port* is the port number for TCP/IP communication. The *port* is always 5450. It is recommended to explicitly define the host and port on the command-line with the **/host** parameter. Nevertheless, if either the host or port is not specified, the interface will attempt to use defaults. Examples: The interface is running on an interface node, the domain name of the PI Data Archive node is Marvin, and the IP address of Marvin is 206.79.198.30. Valid **/host** parameters would be: **/host=marvin** **/host=marvin:5450** **/host=206.79.198.30** **/host=206.79.198.30:5450**
/id=*x* Highly Recommended	The **/id** parameter is used to specify the interface identifier. The interface identifier is a string that is no longer than 9 characters in length. UniInt concatenates this string to the header that is used to identify error messages as belonging to a particular interface. See the Appendix A. Error and Informational Messages for more information. UniInt always uses the **/id** parameter in the fashion described above. This interface also uses the **/id** parameter to identify a particular interface copy number that corresponds to an integer value that is assigned to one of the Location code point attributes, most frequently Location1. For this interface, use only numeric characters in the identifier. For example, **/id=1**
/ps=*x* Required	The **/ps** parameter specifies the point source for the interface. **X** is not case sensitive and can be any single or multiple character string. For example, **/ps=P** and **/ps=p** are equivalent. The length of **X** is limited to 100 characters by UniInt. **X** can contain any character except '*' and '?'. The point source that is assigned with the **/ps** parameter corresponds to the PointSource attribute of individual PI points. The interface will attempt to load only those PI points with the appropriate point source. If the PI API version being used is prior to 1.6.x or the PI Data Archive version is prior to 3.4.370.x, the `PointSource` is limited to a single character unless the SDK is being used.

/stopstat=*digstate* or **/stopstat** **/stopstat** only is equivalent to **/stopstat="***"ntf Shut"*" Optional Default = no digital state written at shutdown.	If **/stopstat=digstate** is present on the command line, then the digital state, `digstate`, will be written to each PI point when the interface is stopped. For a PI Data Archive, `digstate` must be in the system digital state table. . UniInt will use the first occurrence of **digstate** found in the table. If the **/stopstat** parameter is present on the startup command line, then the digital state "`Intf Shut`" will be written to each PI point when the interface is stopped. If neither **/stopstat** nor **/stopstat=digstate** is specified on the command line, then no digital states will be written when the interface is shut down. **Note:** The **/stopstat** parameter is disabled If the interface is running in a UniInt failover configuration as defined in the UniInt Failover Configuration section of this manual. Therefore, the digital state, `digstate`, will not be written to each PI point when the interface is stopped. This prevents the digital state being written to PI points while a redundant system is also writing data to the same PI points. The **/stopstat** parameter is disabled even if there is only one interface active in the failover configuration. Examples: **/stopstat=shutdown** **/stopstat="Intf Shut"** The entire `digstate` value should be enclosed within double quotes when there is a space in `digstate`.
/UFO_ID=# Required for UniInt failover phase 1 or 2	Failover ID. This value must be different from the failover ID of the other interface in the failover pair. It can be any positive, non-zero integer.
/UFO_Interval=# Optional Default: `1000` for phase 1 failover Default: `5000` for phase 2 failover Valid values are 50-20000.	Failover Update Interval Specifies the heartbeat update interval in milliseconds and must be the same on both interface computers. This is the rate at which UniInt updates the failover heartbeat points as well as how often UniInt checks on the status of the other copy of the interface.
/UFO_OtherID=# Required for UniInt failover phase 1 or 2	Other Failover ID. This value must be equal to the failover ID configured for the other interface in the failover pair.

/UFO_Sync=*path/[filename]* Required for UniInt failover phase 2 synchronization. Any valid pathname / any valid filename The default filename is generated as *executablename_pointsource_interfaceID.dat*	The failover synchronization file *path* and optional *filename* specify the path to the shared file used for failover synchronization and an optional filename used to specify a user defined filename in lieu of the default filename. The *path* to the shared file directory can be a fully qualified machine name and directory, a mapped drive letter, or a local path if the shared file is on one of the interface nodes. The *path* must be terminated by a slash (/) or backslash (\ ) character. If no d terminating slash is found, in the **/UFO_Sync** parameter, the interface interprets the final character string as an optional *filename*. The optional *filename* can be any valid filename. If the file does not exist, the first interface to start attempts to create the file. **Note:** If using the optional filename, **do not** supply a terminating slash or backslash character. If there are any spaces in the *path* or *filename*, the entire path and filename must be enclosed in quotes. **Note:** If you use the backslash and path separators and enclose the path in double quotes, the final backslash must be a double backslash (\\). Otherwise the closing double quote becomes part of the parameter instead of a parameter separator. Each node in the failover configuration must specify the same path and filename and must have read, write, and file creation rights to the shared directory specified by the *path* parameter. The service that the interface runs against must specify a valid logon user account under the “Log On” tab for the service properties.
/UFO_Type=*type* Required for UniInt failover phase 2.	The failover Type indicates which type of failover configuration the interface will run. The valid types for failover are HOT, WARM, and COLD configurations. If an interface does not supported the requested type of failover, the interface will shut down and log an error to the log file stating the requested failover type is not supported. This interface supports failover configurations COLD and HOT.

/configfile=<UNC Path> Optional Default: PIC37118.xml	This parameter specifies the interface configuration file to use to define the C37.118 device that the interface will communicate with.
/TagConfig=<UNC Path> Optional Default: None	The `/TagConfig` parameter is used to assist in creating PI tags. Because the C37.118 specification does not dictate the names to use for Phasor, Analog, and Digital channels, different PMU vendors use different names. When the interface is configured initially, the command-line parameter can be specified temporarily in the Additional Parameters test box in the ICU. When the interface is started it reads the CONFIG2 block from the PDC/PMU and generates a comma-delimited file that contains the various PI point attributes. The file can then be edited as required and then imported into PI using SMT. Because the interface exits after the file is written, it is important to remove the command-line parameter for normal operation. If a path is not specified on the command line, the generated file is saved in the interface folder. In versions of the interface earlier than 1.0.4.88, the file was saved in the system drive folder: `Windows\System32`. Optionally, you can specify the fully qualified path of where to save the taglist.csv file; for example: `/TagConfig= C:\Taglist.csv`. If the path or filename contains spaces, you must enclose the path or filename within quotation marks.

Sample PIC37118.bat File

The following is an example file:

```
REM==================================================================
REM
REM PIC37118.bat
REM
REM Sample startup file for the PI Interface for IEEE C37.118
REM
REM==================================================================
REM
REM OSIsoft strongly recommends using PI ICU to modify startup files.
REM
REM Sample command line
REM
    .\PIC37118.exe ^
    /configfile="E:\Program Files\PIPC\Interfaces\C37118\PIC37118.xml" ^
    /ps=C37118 ^
    /ID=1 ^
    /host=XXXXXX:5450
REM
REM End of pic37118.bat File
```

Chapter 9. UniInt Failover Configuration

Introduction

To minimize data loss during a single point of failure within a system, UniInt provides two failover schemes: (1) synchronization through the data source and (2) synchronization through a shared file. Synchronization through the data source is *Phase 1*, and synchronization through a shared file is *Phase 2*.

Phase 1 UniInt Failover is not supported by this interface.

Phase 2 UniInt Failover uses a shared file to synchronize failover operations and provides for *hot*, *warm*, or *cold failover*. The Phase 2 hot failover configuration provides a *no data loss* solution for a single point of failure similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition.

Note: This interface supports only Phase 2 failover, COLD and HOT configurations.

You can also configure UniInt failover to send data to a High Availability (HA) PI Data Archive collective. The collective provides redundant PI Data Archives to allow for the uninterrupted collection and presentation of PI time series data. In an HA configuration, PI Data Archives can be taken down for maintenance or repair. The HA PI Data Archive collective is described in the *High Availability Administrator Guide*.

When configured for UniInt failover, the interface routes all PI data through a state machine. The state machine determines whether to queue data or send it directly to PI depending on the current state of the interface. When the interface is in the active state, data sent through the interface gets routed directly to a PI point. In the backup state, data from the interface gets queued for a short period. Queued data in the backup interface ensures a *no-data loss* failover under normal circumstances for Phase 1 and for the hot failover configuration of Phase 2.

Quick Overview

The Quick Overview below may be used to configure this interface for failover. The failover configuration requires the two copies of the interface participating in failover be installed on different nodes. Users should verify non-failover interface operation as discussed in the Installation Checklist chapter of this manual prior to configuring the interface for failover operations. If you are not familiar with UniInt failover configuration, return to this section after reading the rest of the UniInt Failover Configuration chapter in detail. If a failure occurs at any step below, correct the error and start again at the beginning of step 6 Test in the table below. For the discussion below, the first copy of the interface configured and tested will be considered the primary interface and the second copy of the interface configured will be the backup interface.

Configuration

- One Data Source
- Two interfaces

Prerequisites

- Interface 1 is the Primary interface for collection of PI data from the data source.
- Interface 2 is the Backup interface for collection of PI data from the data source.
- You must set up a shared file if using Phase 2 failover.
- Phase 2: The shared file must store data for five failover tags:

  (1) Active ID.

  (2) Heartbeat 1.

  (3) Heartbeat 2.

  (4) Device Status 1.

  (5) Device Status 2.
- Each interface must be configured with two required failover command line parameters: (1) its FailoverID number (**`/UFO_ID`**); (2) the FailoverID number of its Backup interface (**`/UFO_OtherID`**). You must also specify the name of the PI Data Archive host for exceptions and PI point updates.
- All other configuration parameters for the two interfaces must be identical.

Synchronization through a Shared File (Phase 2)

Figure 1: Synchronization through a Shared File (Phase 2) Failover Architecture

The Phase 2 failover architecture is shown in Figure 2 which depicts a typical network setup including the path to the synchronization file located on a File Server (FileSvr). Other configurations may be supported and this figure is used only as an example for the following discussion.

For a more detailed explanation of this synchronization method, see Detailed Explanation of Synchronization through a Shared File (Phase 2)

Configuring Synchronization through a Shared File (Phase 2)

<table>
<tr><th>Step</th><th>Description</th></tr>
<tr><td>1.</td><td>Verify non-failover interface operation as described in the Installation Checklist section of this manual</td></tr>
<tr><td>2.</td><td>Configure the Shared File
Choose a location for the shared file. The file can reside on one of the interface nodes or on a separate node from the interfaces; however OSIsoft strongly recommends that you put the file on a Windows Server platform that has the “File Server” role configured. .
Setup a file share and make sure to assign the permissions so that both primary and backup interfaces have read/write access to the file.</td></tr>
<tr><td>3.</td><td>Configure the interface parameters
Use the Failover section of the interface Configuration Utility (ICU) to enable failover and create two parameters for each interface: (1) a Failover ID number for the interface; and (2) the Failover ID number for its backup interface.
The Failover ID for each interface must be unique and each interface must know the Failover ID of its backup interface.
If the interface can perform using either Phase 1 or Phase 2 pick the Phase 2 radio button in the ICU.
Select the synchronization File Path and File to use for Failover.
Select the type of failover required (Cold, Warm, Hot). The choice depends on what types of failover the interface supports.
Ensure that the user name assigned in the “Log on as:” parameter in the Service section of the ICU is a user that has read/write access to the folder where the shared file will reside.
All other command line parameters for the primary and secondary interfaces must be identical.
If you use a PI Data Archive Collective, you must point the primary and secondary interfaces to different members of the collective by setting the SDK Member under the PI Host Information section of the ICU.
[Option] Set the update rate for the heartbeat point if you need a value other than the default of 5000 milliseconds.</td></tr>
<tr><td>4.</td><td>Configure the PI points
Configure five PI points for the interface: the active ID, heartbeat 1, heartbeat 2, device status 1 and device status 2. You can also configure two state points for monitoring the status of the interfaces.
Do not confuse the failover device status points with the UniInt health device status points. The information in the two points is similar, but the failover device status points are integer values and the health device status points are string values.
<table>
<tr><th>Tag</th><th>ExDesc</th><th>digitalset</th><th></th></tr>
<tr><td>ActiveID</td><td>[UFO2_ACTIVEID]</td><td></td><td rowspan="7">UniInt does not examine the remaining attributes, but the PointSource and Location1 must match.</td></tr>
<tr><td>IF1_Heartbeat
(IF-Node1)</td><td>[UFO2_HEARTBEAT:#]</td><td></td></tr>
<tr><td>IF2_Heartbeat
(IF-Node2)</td><td>[UFO2_HEARTBEAT:#]</td><td></td></tr>
<tr><td>IF1_DeviceStatus
(IF-Node1)</td><td>[UFO2_DEVICESTAT:#]</td><td></td></tr>
<tr><td>IF2_DeviceStatus
(IF-Node2)</td><td>[UFO2_DEVICESTAT:#]</td><td></td></tr>
<tr><td>IF1_State
(IF-Node1)</td><td>[UFO2_STATE:#]</td><td>IF_State</td></tr>
<tr><td>IF2_State
(IF-Node2)</td><td>[UFO2_STATE:#]</td><td>IF_State</td></tr>
</table></td></tr>
<tr><td>5.</td><td>Test the configuration.</td></tr>
</table>

Step	Description
	After configuring the shared file and the interface and PI points, the interface should be ready to run. For help resolving failover file issues, see OSIsoft KB article - KB00889 - Troubleshooting Failover Synchronization File Errors on the OSIsoft technical support web site. See Troubleshooting UniInt Failover for help resolving Failover issues. 1. Start the primary interface interactively without buffering. 2. Verify a successful interface start by reviewing the log file. The log file will contain messages that indicate the failover state of the interface. A successful start with only a single interface copy running will be indicated by an informational message stating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface not available.`" If the interface has failed to start, an error message will appear in the log file. For details relating to informational and error messages, refer to Appendix A. Error and Informational Messages. 3. Verify data on the PI Data Archive using available PI tools. • The active ID control point in the PI Data Archive must be set to the value of the running copy of the interface as defined by the **`/UFO_ID`** startup command-line parameter. • The heartbeat control point in the PI Data Archive must be changing values at a rate specified by the **`/UFO_Interval`** startup command-line parameter. 4. Stop the primary interface. 5. Start the backup interface interactively without buffering. Notice that this copy will become the primary because the other copy is stopped. 6. Repeat steps 2, 3, and 4. 7. Stop the backup interface. 8. Start buffering. 9. Start the primary interface interactively. 10. Once the primary interface has successfully started and is collecting data, start the backup interface interactively. 11. Verify that both copies of the interface are running in a failover configuration. • Review the log file for the copy of the interface that was started first. The log file will contain messages that indicate the failover state of the interface. The state of this interface must have changed as indicated with an informational message stating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface available.`" If the interface has not changed to this state, browse the log file for error messages. For details relating to informational and error messages, refer to Appendix A. Error and Informational Messages. • Review the log file for the copy of the interface that was started last. The log file will contain messages that indicate the failover state of the interface. A successful start of the interface will be indicated by an informational message stating "`UniInt failover: Interface in the "Backup" state.`" If the interface has failed to start, an error message will appear in the log file. For details relating to informational and error messages, refer to the Appendix A. Error and Informational Messages section below. 12. Verify data in the PI Data Archive using available PI tools.

Step	Description
	• The active ID control point in the PI Data Archive must be set to the value of the running copy of the interface that was started first as defined by the **`/UFO_ID`** startup command-line parameter. • The heartbeat control points for both copies of the interface in the PI Data Archive must be changing values at a rate specified by the **`/UFO_Interval`** startup command-line parameter or the scan class which the points have been built against. 13. Test Failover by stopping the primary interface. 14. Verify the backup interface has assumed the role of primary by searching the `pipc.log` file for a message indicating the backup interface has changed to the "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface not available`." The backup interface is now considered primary and the previous primary interface is now backup. 15. Verify data in the PI Data Archive. For hot failover, there may be an overlap of data due to the queuing of data, but there must be no data loss. For warm or cold failover, short gaps in archived data are expected. 16. Start the backup interface. Once the primary interface detects a backup interface, the primary interface will now change state indicating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface available`." In the `pipc.log` file. 17. Verify the backup interface starts and assumes the role of backup. A successful start of the backup interface will be indicated by an informational message stating "`UniInt failover: Interface in "Backup state.`" Since this is the initial state of the interface, the informational message will be near the beginning of the start sequence of the `pipc.log` file. 18. Test failover with different failure scenarios (for example, loss of the PI Data Archive connection for a single interface copy). UniInt hot failover guarantees no data loss with a single point of failure; verify no data loss by checking the data in the PI Data Archive and on the data source. For warm or cold failover, short gaps in archived data can occur. 19. Stop both copies of the interface, start buffering, start each interface as a service. 20. Verify data as stated above. 21. To designate a specific interface as primary, set the active ID point to the ID of the desired primary interface as defined by the **`/UFO_ID`** startup command-line parameter.

Configuring UniInt Failover through a Shared File (Phase 2)

Start-Up Parameters

Note: The **`/stopstat`** parameter is disabled if the interface is running in a UniInt failover configuration. Therefore, the digital state, `digstate`, will not be written to each PI point when the interface is stopped. This prevents the digital state being written to PI points while a redundant system is also writing data to the same PI points. The **`/stopstat`** parameter is disabled even if there is only one interface active in the failover configuration.

The following table lists the start-up parameters used by UniInt Failover Phase 2. All of the parameters are required except the **`/UFO_Interval`** startup parameter. See the table below for further explanation.

Parameter	Required/ Optional	Description	Value/Default
`/UFO_ID=`*`#`*	Required	Failover ID for IF-Node1 This value must be different from the failover ID of IF-Node2.	Any positive, non-zero integer / **1**
	Required	Failover ID for IF-Node2 This value must be different from the failover ID of IF-Node1.	Any positive, non-zero integer / **2**
`/UFO_DNFBPI`	Optional	If this parameter is enabled and both the primary and backup instances lose connection to the PI Data Archive UniInt will not failover and incoming data will be buffered by a single interface instance to prevent the buffer from being split.	Undefined
`/UFO_OtherID=`*`#`*	Required	Other Failover ID for IF-Node1 The value must be equal to the Failover ID configured for the interface on IF-Node2.	Same value as Failover ID for IF-Node2 / **2**
	Required	Other Failover ID for IF-Node2 The value must be equal to the Failover ID configured for the interface on IF-Node1.	Same value as Failover ID for IF-Node1 / **1**
`/UFO_Sync=` *`path/[filename]`*	Required for Phase 2 synchronization	The Failover File Synchronization file *path* and optional *filename* specify the path to the shared file used for failover synchronization and an optional filename used to specify a user defined filename in lieu of the default filename. The *path* to the shared file directory can be a fully qualified machine name and directory, a mapped drive letter, or a local path if the shared file is on one of the interface nodes. The *path* must be terminated by a slash (/) or backslash (\ ) character.	Any valid pathname / any valid filename The default filename is generated as *executablename_ pointsource_ interfaceID.dat*

Parameter	Required/ Optional	Description	Value/Default
		If no terminating slash is found, in the **`/UFO_Sync`** parameter, the interface interprets the final character string as an optional *filename*. The optional *filename* can be any valid filename. If the file does not exist, the first interface to start attempts to create the file. **Note:** If using the optional filename, **do not** supply a terminating slash or backslash character. If there are any spaces in the *path* or *filename*, the entire path and filename must be enclosed in quotes. **Note:** If you use the backslash and path separators and enclose the path in double quotes, the final backslash must be a double backslash (`\\`). Otherwise the closing double quote becomes part of the parameter instead of a parameter separator. Each node in the failover configuration must specify the same path and filename and must have read, write, and file creation rights to the shared directory specified by the *path* parameter. The service that the interface runs against must specify a valid logon user account under the “Log On” tab for the service properties.	
`/UFO_Type=`*`type`*	Required	The Failover Type indicates which type of failover configuration the interface will run. The valid types for failover are HOT, WARM, and COLD configurations. If an interface does not supported the requested type of failover, the interface will shutdown and log an error to the `pipc.log` file stating the requested failover type is not supported.	COLD|WARM|HOT / **COLD**

Parameter	Required/ Optional	Description	Value/Default
/UFO_Interval=*#*	Optional	Failover Update Interval Specifies the heartbeat Update Interval in milliseconds and must be the same on both interface computers. This is the rate at which UniInt updates the failover heartbeat points as well as how often UniInt checks on the status of the other copy of the interface.	50 – 20000 / **5000**
/Host=*server*	Required	Host PI Data Archive for exceptions and PI point updates The value of the **/Host** startup parameter depends on the PI Data Archive configuration. If the PI Data Archive is not part of a collective, the value of **/Host** must be identical on both interface computers. If the redundant interfaces are being configured to send data to a PI Data Archive collective, set the value of the **/Host** parameters on the different interface nodes to different members of the collective.	For IF-Node1 PrimaryPI / **None** For IF-Node2 SecondaryPI / **None**

Failover Control Points

The following table describes the points that are required to manage failover. In Phase 2 Failover, these points are located in a data file shared by the primary and backup interfaces.

OSIsoft recommends that you locate the shared file on a dedicated server that has no other role in data collection. This avoids potential resource contention and processing degradation if your system monitors a large number of data points at a high frequency.

Point	Description	Value / Default
ActiveID	Monitored by the interfaces to determine which interface is currently sending data to the PI Data Archive. **ActiveID** must be initialized so that when the interfaces read it for the first time, it is not an error. **ActiveID** can also be used to force failover. For example, if the current primary is IF-Node 1 and **ActiveID** is 1, you can manually change **ActiveID** to 2. This causes the interface at IF-Node2 to transition to the primary role and the interface at IF-Node1 to transition to the backup role.	From 0 to the highest interface Failover ID number / **None**) Updated by the redundant interfaces Can be changed manually to initiate a manual failover
Heartbeat 1	Updated periodically by the interface on IF-Node1. The interface on IF-Node2 monitors this value to determine if the interface on IF-Node1 has become unresponsive.	Values range between 0 and 31 / **None** Updated by the interface on IF-Node1

Point	Description	Value / Default
Heartbeat 2	Updated periodically by the interface on IF-Node2. The interface on IF-Node1 monitors this value to determine if the interface on IF-Node2 has become unresponsive.	Values range between 0 and 31 / **None** Updated by the interface on IF-Node2

PI Points

The following tables list the required UniInt failover control PI points, the values they will receive, and descriptions.

Active_ID Point Configuration

Attributes	ActiveID
Tag	**<Intf>_ActiveID**
CompMax	**0**
ExDesc	[UFO2_ActiveID]
Location1	Match # in **/id=#**
Location5	Optional, Time in min to wait for backup to collect data before failing over.
PointSource	Match *x* in **/ps=***x*
PointType	Int32
Shutdown	0
Step	1

Heartbeat and Device Status Point Configuration

Attribute	Heartbeat 1	Heartbeat 2	DeviceStatus 1	DeviceStatus 2
Tag	<HB1>	<HB2>	<DS1>	<DS2>
ExDesc	[UFO2_Heartbeat: #] Match # in **/UFO_ID=#**	[UFO2_Heartbeat: #] Match # in **/UFO_OtherID=#**	[UFO2_DeviceStat: #] Match # in **/UFO_ID=#**	[UFO2_DeviceStat: #] Match # in **/UFO_OtherID=#**
Location1	Match # in **/id=#**	Match # in **/id=#**	Match # in **/id=#**	Match # in **/id=#**
Location5	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.
Point Source	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*
PointType	int32	int32	int32	int32
Shutdown	0	0	0	0
Step	1	1	1	1

Interface State Point Configuration

Attribute	Primary	Backup
Tag	<Tagname1>	<Tagname2>
CompMax	0	0

Attribute	Primary	Backup
DigitalSet	UFO_State	UFO_State
ExDesc	[UFO2_State:#] (Match **/UFO_ID=**# on primary node)	[UFO2_State:#] (Match **/UFO_ID=**# on backup node)
Location1	Match # in **/id=**#	Same as for primary node
PointSource	Match *x* in **/ps=***x*	Same as for primary node
PointType	digital	digital
Shutdown	0	0
Step	1	1

The following table describes the extended descriptor for the above PI points in more detail.

PI Tag ExDesc	Required / Optional	Description	Value
[UFO2_ACTIVEID]	Required	Active ID point The ExDesc must start with the case sensitive string: [UFO2_ACTIVEID]. The PointSource must match the interfaces' Pointsource. Location1 must match the ID for the interfaces. Location5 is the COLD failover retry interval in minutes. This can be used to specify how long before an interface retries to connect to the device in a COLD failover configuration. (See the description of COLD failover retry interval for a detailed explanation.)	0 – highest Interface Failover ID Updated by the redundant interfaces
[UFO2_HEARTBEAT:#] (IF-Node1)	Required	Heartbeat 1 point The ExDesc must start with the case sensitive string: [UFO2_HEARTBEAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node1. The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces.	0 – 31 / **None** Updated by the interface on IF-Node1
[UFO2_HEARTBEAT:#] (IF-Node2)	Required	Heartbeat 2 point The ExDesc must start with the case sensitive string: [UFO2_HEARTBEAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2. The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces.	0 – 31 / **None** Updated by the interface on IF-Node2

PI Tag ExDesc	Required / Optional	Description	Value
[UFO2_DEVICESTAT :#] (IF-Node1)	Required	Device Status 1 Tag The ExDesc must start with the case sensitive string: [UFO2_DEVICESTAT:#] The value following the colon (☺ must be the failover ID for the interface running on IF-Node1 The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces. A lower value is a better status and the interface with the lower status will attempt to become the primary interface. The failover 1 device status point is very similar to the UniInt Health Device Status point except the data written to this point are integer values. A value of 0 is good and a value of 99 is OFF. Any value between these two extremes may result in a failover. The interface client code updates these values when the health device status point is updated.	0 – 99 / **None** Updated by the interface on IF-Node1
[UFO2_DEVICESTAT :#] (IF-Node2)	Required	Device Status 2 point The ExDesc must start with the case sensitive string: [UFO2_DEVICESTAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2 The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces. A lower value is a better status and the interface with the lower status will attempt to become the primary interface.	0 – 99 / **None** Updated by the interface on IF-Node2
[UFO2_STATE:#] (IF-Node1)	Optional	State 1 point The ExDesc must start with the case sensitive string: [UFO2_STATE:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node1 The failover state point is recommended. The failover state points are digital points assigned to a digital state set with the following values. 0 = Off: The interface has been shut down.	0 – 5 / **None** Normally updated by the interface currently in the primary role.

PI Tag ExDesc	Required / Optional	Description	Value
		1 = Backup No Data Source: The interface is running but cannot communicate with the data source. 2 = Backup No PI Connection: The interface is running and connected to the data source but has lost its communication to the PI Data Archive. 3 = Backup: The interface is running and collecting data normally and is ready to take over as primary if the primary interface shuts down or experiences problems. 4 = Transition: The interface stays in this state for only a short period of time. The transition period prevents thrashing when more than one interface attempts to assume the role of primary interface. 5 = Primary: The interface is running, collecting data and sending the data to the PI Data Archive.	
[UFO2_STATE:#] (IF-Node2)	Optional	State 2 point The ExDesc must start with the case sensitive string: [UFO2_STATE:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2 The failover state point is recommended.	Normally updated by the interface currently in the Primary state. Values range between 0 and 5. See description of State 1 point.

Detailed Explanation of Synchronization through a Shared File (Phase 2)

In a shared file failover configuration, there is no direct failover control information passed between the data source and the interface. This failover scheme uses five PI tags to control failover operation, and all failover communication between primary and backup interfaces passes through a shared data file.

Once the interface is configured and running, the ability to read or write to the PI tags is not required for the proper operation of failover (unless connection to the shared file is lost). This solution does not require a connection to the PI Data Archive after initial startup because the control point data are set and monitored in the shared file. However, the PI point values are sent to the PI Data Archive so that you can monitor them with standard OSIsoft client tools.

You can force manual failover by changing the **ActiveID** on the PI Data Archive to the backup failover ID.

The preceding figure shows a typical network setup in the normal or steady state. The solid magenta lines show the data path from the interface nodes to the shared file used for failover synchronization. The shared file can be located anywhere in the network as long as both interface nodes can read, write, and create the necessary file on the shared file machine. OSIsoft strongly recommends that you put the file on a dedicated file server that has no other role in the collection of data.

The major difference between synchronizing the interfaces through the data source (Phase 1) and synchronizing the interfaces through the shared file (Phase 2) is where the control data is located. When synchronizing through the data source, the control data is acquired directly from the data source. We assume that if the primary interface cannot read the failover control points, then it cannot read any other data. There is no need for a backup communications path between the control data and the interface.

When synchronizing through a shared file, however, phase 2 cannot assume that loss of control information from the shared file implies that the primary interface is down. We must account for the possible loss of the path to the shared file itself and provide an alternate control path to determine the status of the primary interface. For this reason, if the shared file is unreachable for any reason, the interfaces use the PI Data Archive as an alternate path to pass control data.

When the backup interface does not receive updates from the shared file, it cannot tell definitively why the primary is not updating the file, whether the path to the shared file is down, whether the path to the data source is down, or whether the interface itself is having problems. To resolve this uncertainty, the backup interface uses the path to the PI Data Archive to determine the status of the primary interface. If the primary interface is still communicating with the PI Data Archive, then failover to the backup is not required. However, if the primary interface is not posting data to the PI Data Archive, then the backup must initiate failover operations.

The primary interface also monitors the connection with the shared file to maintain the integrity of the failover configuration. If the primary interface can read and write to the shared file with no errors but the backup control information is not changing, then the backup is experiencing some error condition. To determine exactly where the problem exists, the primary interface uses the path to the PI Data Archive to establish the status of the backup interface. For example, if the backup interface information indicates that it has been shutdown, it may have been restarted and is now experiencing errors reading and writing to the shared file. Both primary and backup interfaces must always check their status through the PI Data Archive to determine if one or the other is not updating the shared file and why.

Steady State Operation

Steady state operation is considered the normal operating condition. In this state, the primary interface is actively collecting data and sending its data to PI points. The primary interface is also updating its heartbeat value, monitoring the heartbeat value for the backup interface, checking the active ID value, and checking the device status for the backup interface every failover update interval on the shared file. Likewise, the backup interface is updating its heartbeat value, monitoring the heartbeat value for the primary interface, checking the active ID value, and checking the device status for the primary interface every failover update interval on the shared file. As long as the heartbeat value for the primary interface indicates that it is operating properly, the active ID has not changed, and the device status on the primary interface is good, the backup interface will continue in this mode of operation.

An interface configured for hot failover will have the backup interface actively collecting and queuing data but not sending that data to the PI Data Archive. An interface for warm failover in the backup role is not actively collecting data from the data source even though it may be configured with PI points and may even have a good connection to the data source. An interface configured for cold failover in the backup role is not connected to the data source and upon initial startup will not have configured PI points.

The interaction between the interface and the shared file is fundamental to failover. The discussion that follows only refers to the data written to the shared file. However, every value written to the shared file is echoed to the points in the PI Data Archive. Updating of the points in the PI Data Archive is assumed to take place unless communication with the PI Data Archive is interrupted. The updates to the PI Data Archive will be buffered by Bufserv or PIBufss in this case.

In a hot failover configuration, each interface participating in the failover solution will queue three failover intervals worth of data to prevent any data loss. When a failover occurs, there may be a period of overlapping data for up to three intervals. The exact amount of overlap is determined by the timing and the cause of the failover and may be different every time. Using the default update interval of 5 seconds will result in overlapping data between 0 and 15 seconds. The no data loss claim for hot failover is based on a single point of failure. If both interfaces have trouble collecting data for the same period of time, data will be lost during that time.

As mentioned above, each interface has its own heartbeat value. In normal operation, the heartbeat value in the shared file is incremented by UniInt from 1 to 15 and then wraps around to a value of 1 again. UniInt increments the heartbeat value in the shared file every failover update interval. The default failover update interval is 5 seconds. UniInt also reads the heartbeat value for the other interface copy participating in failover every failover update interval. If the connection to the PI Data Archive is lost, the value of the heartbeat will be incremented from 17 to 31 and then wrap around to a value of 17 again. Once the connection to the PI Data Archive is restored, the heartbeat values will revert back to the 1 to 15 range. During a normal shutdown process, the heartbeat value will be set to zero.

During steady state, the active ID will equal the value of the failover ID of the primary interface. This value is set by UniInt when the interface enters the primary state and is not updated again by the primary interface until it shuts down gracefully. During shutdown, the primary interface will set the active ID to zero before shutting down. The backup interface has the ability to assume control as primary even if the current primary is not experiencing problems. This can be accomplished by setting the active ID point in the PI Data Archive to the active ID of the desired interface copy.

As previously mentioned, in a hot failover configuration the backup interface actively collects data but does not send its data to PI points. To eliminate any data loss during a failover, the backup interface queues data in memory for three failover update intervals. The data in the queue is continuously updated to contain the most recent data. Data older than three update intervals is discarded if the primary interface is in a good status as determined by the backup. If the backup interface transitions to the primary, it will have data in its queue to send to the PI points. This queued data is sent to the PI points using the same function calls that would have been used had the interface been in a primary state when the data was collected. If UniInt receives data without a timestamp, the primary copy uses the current PI time to timestamp data sent to PI points. Likewise, the backup copy timestamps data it receives without a timestamp with the current PI time before queuing its data. This preserves the accuracy of the timestamps.

Failover Configuration Using PI ICU

The use of the PI ICU is the recommended and safest method for configuring the interface for UniInt failover. With the exception of the notes described in this section, the interface shall be configured with the PI ICU as described in the Configuring the Interface with PI ICU section of this manual.

Note: With the exception of the **`/UFO_ID`** and **`/UFO_OtherID`** startup command-line parameters, the UniInt failover scheme requires that both copies of the interface have identical startup command files. This requirement causes the PI ICU to produce a message when creating the second copy of the interface stating that the "PS/ID combo is already in use by the interface" as shown in

Figure below. Ignore this message and click the *Add* button.

Create the Interface Instance with PI ICU

If the interface does not already exist in the ICU it must first be created. The procedure for doing this is the same as for non-failover interfaces. When configuring the second instance for UniInt Failover the *Point Source* and *Interface ID #* boxes will be in yellow and a message will be displayed saying this is already in use. This should be ignored.

Figure 4: PI ICU configuration screen shows that the "PS/ID combo is already in use by the interface." The user must ignore the yellow boxes, which indicate errors, and click the *Add* button to configure the interface for failover.

Configuring the UniInt Failover Startup Parameters with PI ICU

There are three interface startup parameters that control UniInt failover: **`/UFO_ID`**, **`/UFO_OtherID`**, and **`/UFO_Interval`**. The `UFO` stands for UniInt Failover. The **`/UFO_ID`** and **`/UFO_OtherID`** parameters are required for the interface to operate in a failover configuration, but the **`/UFO_Interval`** is optional. For more detail on each of these parameters, see the Configuring UniInt Failover through a Shared File (Phase 2) Start-Up Parameters section of this manual.

Figure 5: This figure illustrates the PI ICU failover configuration screen showing the UniInt failover startup parameters (Phase 2). This copy of the interface defines its Failover ID as 2 (**`/UFO_ID=2`**) and the other interfaces Failover ID as 1 (**`/UFO_OtherID=1`**). The other failover interface copy must define its Failover ID as 1 (**`/UFO_ID=1`**) and the other interface Failover ID as 2 (**`/UFO_OtherID=2`**) in its ICU failover configuration screen. It also defines the location and name of the synchronization file as well as the type of failover as COLD.

Creating the Failover State Digital State Set

The UFO_State digital state set is used in conjunction with the failover state digital point. If the UFO_State digital state set has not been created yet, it can be created using either the *Failover* page of the ICU (1.4.1.0 or later) or the Digital States plug-in in the SMT 3 Utility (3.0.0.7 or later).

Using the PI ICU Utility to create Digital State Set

To use the UniInt *Failover* page to create the UFO_State digital state set, right-click on any of the failover points listed in the *Tag* column and then click the *Create UFO_State Digital Set on Server XXXXXX…* command, where *XXXXXX* is the PI Data Archive where the points will be or are created.

This command is unavailable if the UFO_State digital state set already exists on the *XXXXXX* PI Data Archive.

Using the PI SMT 3 Utility to create Digital State Set

Optionally the *Export UFO_State Digital Set (.csv)* command on the shortcut menu can be selected to create a comma-separated file to be imported via the System Management Tools (SMT3) (version 3.0.0.7 or higher) or use the `UniInt_Failover_DigitalSet_UFO_State.csv` file included in the installation kit.

The procedure below outlines the steps necessary to create a digital set on a PI Data Archive using the *Import from File* command found in the SMT3 application. The procedure assumes the user has a basic understanding of the SMT3 application.

1. Open the SMT3 application.
2. Select the appropriate PI Data Archive from the PI Data Archives window. If the desired server is not listed, add it using the PI Connection Manager. A view of the SMT application is shown in Figure 7 below.
3. From the *System Management Plug-Ins* window, expand *Points* then select *Digital States*. A list of available digital state sets will be displayed in the main window for the selected PI Data Archive. Refer to Figure 7 below.
4. In the main window, right-click on the desired server and select the *Import from File* command. Refer to Figure 7 below.

Figure 7: PI SMT application configured to import a digital state set file. The PI Data Archives window shows the "localhost" PI Data Archive selected along with the System Management Plug-Ins window showing the Digital States Plug-In as being selected. The digital state set file can now be imported by selecting the *Import from File* command.

5. Navigate to and select the `UniInt_Failover_DigitalSet_UFO_State.csv` file for import using the Browse icon on the display. Select the desired *Overwrite Options*. Refer to Figure below.

Figure 8: PI SMT application *Import Digital Set(s)* window. This view shows the `UniInt_Failover_DigitalSet_UFO_State.csv` file as being selected for import. Select the desired *Overwrite Options* by choosing the appropriate option button.

6. Click on the *OK* button. Refer to Figure above.

7. The `UFO_State` digital set is created as shown in Figure below.

Figure 9: The PI SMT application showing the `UFO_State` digital set created on the "localhost" PI Data Archive.

Creating the UniInt Failover Control and Failover State Points (Phase 2)

The ICU can be used to create the UniInt failover control and state points.

To use the ICU *Failover* page to create these points, simply right-click any of the failover points in the list and click the *Create all points (UFO Phase 2)* command.

If this menu command is unavailable, it is because the UFO_State digital state set has not been created in the PI Data Archive yet. *Create UFO_State Digital Set on Server xxxxxxx…* on the shortcut menu can be used to create that digital state set. After this has been done then the *Create all points (UFO Phase2)* command should be available.

Once the failover control and failover state tags have been created the *Failover* page of the ICU should look similar to the illustration below.

Chapter 10. Interface Node Clock

Make sure that the time and time zone settings on the computer are correct. To confirm, run the Date/Time applet located in the Windows Control Panel. If the locale where the interface node resides observes Daylight Saving Time, check the *Automatically adjust clock for daylight saving changes* box. For example,

In addition, make sure that the TZ environment variable is not defined. All of the currently defined environment variables can be viewed by opening a Command Prompt window and typing `set`. That is,

```
C:> set
```

Confirm that TZ is not in the resulting list. If it is, run the System applet of the Control Panel, click the *Environment Variables* button under the *Advanced* tab, and remove TZ from the list of environment variables.

Chapter 11. Security

Windows

The PI Firewall Database and the PI Trust Database must be configured so that the interface is allowed to write data to the PI Data Archive.

The Trust Database, which is maintained by the PI Base Subsystem, replaces the Proxy Database used prior to PI Data Archive version 3.3. The PI Trust Database maintains all the functionality of the proxy mechanism while being more secure.

See "Manage Interface Authentication with PI Trusts" in the chapter "Manage Security" of the *PI Server Introduction to System Management Guide*.

If the interface cannot write data to the PI Data Archive because it has insufficient privileges, a `-10401` error will be reported in the log file. If the interface cannot send data to a PI2 Data Archive, it writes a -999 error. See the section Appendix A. Error and Informational Messages for additional information on error messaging.

Authentication

Interface instances are usually configured to run as Windows services. Since a service runs in a non-interactive context, a PI trust is required to authenticate the interface service to the PI Data Archive. A PI trust is associated with one PI identity, PI user, or PI group. When an interface successfully authenticates through a trust, the interface is granted the access rights for the associated identity, user, or group.

OSIsoft discourages using highly-privileged identities, users, or groups in PI trusts for interfaces.

Security Note: Avoid using the piadmin super-user and piadmins group. The recommended best practice for PI Data Archive security is to create an identity, user, or group that has only the access rights which are necessary for the interface to operate.

PI Data Archive v3.3 and Later

Security Configuration using Trust Editor

The Trust Editor plug-in for PI System Management Tools edits the PI trust table.

See the "Manage Interface Authentication with PI Trusts" section in the PI Server Introduction to System Management manual for more details on security configuration.

Security configuration using piconfig

For PI Data Archive v3.3 and higher, the following example demonstrates how to edit the PI trust table with ipconfig.

```
C:\PI\adm> ipconfig
@table pitrust
@mode create
@istr Trust,IPAddr,NetMask,PIUser
a_trust_name,192.168.100.11,255.255.255.255,trust_identity
@quit
```

For the preceding example,

`Trust`: An arbitrary name for the trust table entry; in the above example,

`a_trust_name`

`IPAddr`: the IP address of the computer running the interface; in the above example,

`192.168.100.11`

`NetMask`: the network mask; *`255.255.255.255`* specifies an exact match with `IPAddr`

`PIUser`: the PI identity, user, or group the interface is entrusted as; in the example,

`trust_identity`

PI Data Archive v3.2

For PI Data Archive v3.2, the following example demonstrates how to edit the PI Proxy table:

```
C:\PI\adm> p100ointed100@table pi_gen,piproxy
@mode create
@istr host,proxyaccount
piapimachine,piadmin
@quit
```

In place of piapimachine, put the name of the interface node as it is seen by the PI Data Archive.

Authorization

For an interface instance to start and write data to PI points, the following permissions must be granted to the PI identity, user, or group in the PI trust that authenticates the interface instance.

Database Security	Permission	Notes
PIPOINT	r	

Point Database	Permission	Notes
PtSecurity	r	
DataSecurity	r,w	Unbuffered
	r	Buffered (the buffering application requires r,w for the interface points)

The permissions in the preceding table must be granted for every PI point that is configured for the interface instance. Observe that buffering on the interface node is significant to PI point permissions.

When the interface instance is running on an unbuffered interface node, the interface instance sends PI point updates directly to the PI Data Archive. Therefore, the DataSecurity attribute must grant write access to the PI identity, user, or group in the PI trust that authenticates the interface instance.

When the interface instance is running on a buffered interface node, the interface instance sends PI point updates to the local buffering application, which relays the PI point updates to the PI Data Archive. The buffering application is a separate client to the PI Data Archive and, therefore, authenticates independently of the interface instances. The DataSecurity attribute must grant write access to the PI identity, user, or group in the PI trust that authenticates the buffering application.

Chapter 12. Starting / Stopping the Interface

This section describes starting and stopping the interface once it has been installed as a service. See the *PI Universal Interface (UniInt) User Guide* to run the interface interactively.

Starting Interface as a Service

If the interface was installed as service, it can be started from PI ICU, the Services control panel or with the command:

```
PIC37118.exe / start [ /serviceid id ]
```

To start the interface service with PI ICU, use the button on the PI ICU toolbar.

A message will inform the user of the status of the interface service. Even if the message indicates that the service has started successfully, double check through the Services control panel applet. Services may terminate immediately after startup for a variety of reasons, and one typical reason is that the service is not able to find the command-line parameters in the associated .bat file. Verify that the root name of the .bat file and the .exe file are the same, and that the .bat file and the .exe file are in the same directory. Further troubleshooting of services might require consulting the log file, Windows Event Viewer, or other sources of log messages. See the section Appendix A. Error and Informational Messages for additional information.

Stopping Interface Running as a Service

If the interface was installed as service, it can be stopped at any time from PI ICU, the Services control panel or with the command:

```
PIC37118.exe /stop [ /serviceid id ]
```

The service can be removed by:

```
PIC37118.exe /remove [ /serviceid id ]
```

To stop the interface service with PI ICU, use the button on the PI ICU toolbar.

Chapter 13. Buffering for PI Interfaces

The interface node uses buffering to prevent data loss when PI Data Archive is not available. UniInt interfaces can buffer data to store point values when network communication to the PI Data Archive is unavailable. UniInt disconnected startup requires buffering, and it is highly recommended for failover. Buffering for interfaces is configured and enabled through PI ICU.

Buffering services

The PI System offers two services to implement buffering at interfaces:

- PI Buffer Subsystem (PIBufss)
- API Buffer Server (Bufserv)

PI Buffer Subsystem is the best option for most environments.

Use API Buffer Server only if one or more of the following conditions is true:

- The version of PI Data Archive receiving the buffered data is earlier than 3.4.375
- Your interfaces run on a non-Windows platform

If any of the above conditions apply to you, see the *PI Buffering Manager Help (PIPC/HELP/BufferManager.chm)* documentation for PI Server. Otherwise, use PI Buffer Subsystem.

Buffering and collectives

PI Buffer Subsystem 4.3 and later can buffer data to multiple independent servers, including those configured as PI Server collectives. For interfaces to use PI Buffer Subsystem with PI Server collectives, the PI Data Archive servers must be running PI Data Archive version 3.4.375 or later.

Caution:

API Buffer Server does not detect and validate the PI Server collective configuration. As a result, API Buffer Server requires manual configuration whenever a collective changes. Also, because API Buffer Server results in compression at the PI Data Archive machine, archives at different servers in the collective could contain different records.

Buffering configuration

Use PI Interface Configuration Utility (PI ICU) to configure interface buffering.

The **Tools** > **Buffering** option helps you configure buffering. Depending on your current configuration, this option does one of the following:

- If this computer is configured to buffer data using PI Buffer Subsystem 4.3 or later, the Buffering Manager window opens and shows a buffering dashboard. The dashboard shows information about the status of buffering on this computer.
- If this computer is not currently configured to buffer data, and PI Buffer Subsystem 4.3 or later is installed, you are prompted to configure PI Buffer Subsystem. If you click **Yes**, the Buffering Manager window opens and shows the installation wizard, which helps you configure PI Buffer Subsystem.
- If this computer is configured to buffer data using API Buffer Server (Bufserv), and PI Buffer Subsystem 4.3 or later is installed, you are prompted to convert to and configure PI Buffer Subsystem. If you click Yes at both prompts, the Buffering Manager window opens and shows the upgrade wizard, which helps you upgrade from API Buffer Server to PI Buffer Subsystem.
- If PI Buffer Subsystem 4.3 is not yet installed, the Buffering window opens for API Buffering or the earlier version of PI Buffer Subsystem.

For PI Buffer Subsystem 4.3, when configuring an interface to buffer data to a PI Data Archive server which has not been added to the buffered server list you must enable buffering. Click the **Enable** button in the **Buffering Status** box on the interface General page. To verify that the buffering status is `On`, exit PI ICU, then restart and select the interface.

You can use Buffering Manager to configure, monitor, and troubleshoot buffering using PI Buffer Subsystem. PI Buffer Subsystem is recommended for applications connecting to PI Data Archive 3.4.375 or later. Older versions of PI Data Archive require API Buffer Server, as do some sites with custom solutions. See Buffering Manager help *(PIPC/HELP/BufferManager.chm)* for more information.

Chapter 14. Interface Diagnostics Configuration

The PI Point Configuration chapter provides information on building PI points for collecting data from the device. This chapter describes the configuration of points related to interface diagnostics.

Note: The procedure for configuring interface diagnostics is not specific to this interface. Thus, for simplicity, the instructions and screenshots that follow refer to an interface named **ModbusE**.

Some of the points that follow refer to a "performance summary interval". This interval is 8 hours by default. You can change this parameter via the *Scan performance summary* box in the *UniInt – Debug* parameter category page:

Scan Class Performance Points

A scan class performance point measures the amount of time (in seconds) that this interface takes to complete a scan. The interface writes this scan completion time to millisecond resolution. Scan completion times close to 0 indicate that the interface is performing optimally. Conversely, long scan completion times indicate an increased risk of missed or skipped scans. To prevent missed or skipped scans, you should distribute the data collection points among several scan classes.

You configure one scan class performance point for each scan class in this interface. From the ICU, select this interface from the *Interface* drop-down list and click *UniInt-Performance Points* in the parameter category pane:

Right click the row for a particular *Scan Class #* to open the shortcut menu:

You need not restart the interface for it to write values to the scan class performance points.

To see the current values (snapshots) of the scan class performance points, right-click and select *Refresh Snapshots*.

Create / Create All

To create a performance point, right-click the line belonging to the point to be created, and select *Create*. Click *Create All* to create all the scan class performance points.

Delete

To delete a performance point, right-click the line belonging to the point to be deleted and select *Delete*.

Correct / Correct All

If the "Status" of a point is marked "Incorrect", the point configuration can be automatically corrected by the ICU by right-clicking on the line belonging to the point to be corrected and selecting *Correct*. The performance points are created with the following PI attribute values.

Attribute	Details
Tag	Point name that appears in the list box
Point Source	Point Source for points for this interface, as specified on the *General* page
Compressing	Off
Excmax	0
Descriptor	*Interface name* + " Scan Class # Performance Point"

If the ICU detects that a performance point is not defined with the correct attributes, the point will be marked *Incorrect*: To correct all points, click *Correct All*.

Rename

Right-click the line belonging to the point and select *Rename* to rename the performance point.

Column descriptions

Status

The *Status* column in the performance points table indicates whether the performance point exists for the scan class in the *Scan Class #* column.

Created – Indicates that the performance point does exist

Not Created – Indicates that the performance point does not exist

Deleted – Indicates that a performance point existed but was just deleted by the user

Scan Class #

The *Scan Class* column indicates to which scan class the performance point in the *Tagname* column belongs. There will be one scan class in the *Scan Class* column for each scan class listed in the *Scan Classes* box on the *General* page.

Tagname

The *Tagname* column holds the performance point name.

PS

This is the point source used for these performance points and the interface.

Location1

This is the value used by the interface for the **`/ID=`**# point attribute.

ExDesc

This is the used to tell the interface that these are performance points and the value is used to correspond to the **`/ID=`**# command line parameter if multiple copies of the same interface are running on the interface node.

Snapshot

The *Snapshot* column holds the snapshot value of each performance point that exists in the PI Data Archive. The *Snapshot* column is updated when the *Performance Points* page is selected, and when the interface is first loaded. You may have to scroll to the right to see the snapshots.

Performance Counters Points

Installing the interface as a Windows service creates Windows performance counters to report performance data. Such data include items like:

- the amount of time that the interface has been running;
- the number of points the interface has added to its point list;
- the number of points that are currently updating among others

There are two types or instances of performance counters that can be collected and stored in PI points. The first is (_Total) which is a total for the performance counter since the interface instance was started. The other is for individual scan classes (Scan Class *x*) where *x* is a particular scan class defined for the interface instance that is being monitored.

OSIsoft's PI Interface for Performance Monitor is capable of reading these performance values and writing them to PI points. Please see the *PI Interface for Performance Monitor* for more information.

If there is no PI Interface for Performance Monitor registered with the ICU in the Module Database for the PI Data Archive the interface is sending its data to, you cannot use the ICU to create any interface instance's performance counters points:

After installing the PI Interface for Performance Monitor as a service, select this interface instance from the *Interface* drop-down list. Click *Performance Counters* in the parameter categories pane and then right-click on the row containing the performance counter point you wish to create. This will open the shortcut menu:

Click *Create* to create the performance counter point for that particular row. Click *Create All* to create all the performance counters points listed which have a status of Not Created.

To see the current values (snapshots) of the created performance counters points, right-click on any row and select *Refresh Snapshots*.

Note: The PI Interface for Performance Monitor – and not this interface – is responsible for updating the values for the performance counters points in the PI Data Archive. So, make sure that the PI Interface for Performance Monitor is running correctly.

Performance Counters

In the following lists of performance counters, the naming convention is:

“PerformanceCounterName” (.PerformanceCounterPointSuffix)

The name created by the ICU for each performance counter point is based on the setting found under the Tools ➔ Options ➔ Naming Conventions ➔ Performance Counter Points. The default for this is “sy.perf.[machine].[if service]” followed by the performance counter point suffix.

Performance Counters for both (_Total) and (Scan Class x)

“Point Count” (.point_count)

A .sched_scans_%missed performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%missed performance counter point indicates the percentage of scans the interface missed per scan class or the total number missed for all scan classes since startup. A missed scan occurs if the interface performs the scan one second later than scheduled.

The ICU uses a naming convention such that the point containing “(Scan Class 1)” (for example, "“`y.perf.etamp390.E1(Scan Class 1).point_count`"” refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing “(_Total)” refers to the sum of all scan classes.

“Scheduled Scans: % Missed” (.sched_scans_%missed)

A .sched_scans_%skipped performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%skipped performance counter point indicates the percentage of scans the interface skipped per scan class or the total number skipped for all scan classes since startup. A skipped scan is a scan that occurs at least one scan period after its scheduled time. This point is similar to the [UI_SCSKIPPED] health point.

The ICU uses a naming convention such that the point containing "“Scan Class 1)" (for example, "“`y.perf.etamp390.E1(Scan Class 1).sched_scans_%missed`"” refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing "“_Total)"”refers to the sum of all scan classes.

“Scheduled Scans: % Skipped” (.sched_scans_%skipped)

A .sched_scans_%skipped performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%skipped performance counter point indicates the percentage of scans the interface skipped per scan class or the total number skipped for all scan classes since startup.

A skipped scan is a scan that occurs at least one scan period after its scheduled time. This point is similar to the [UI_SCSKIPPED] health point.

The ICU uses a naming convention such that the point containing "“Scan Class 1)" (for example, "`sy.perf.etamp390.E1(Scan Class 1).sched_scans_%skipped`") refers to scan class 1, "“Scan Class 2)"”refers to scan class 2, and so on. The point containing "“_Total)"”refers to the sum of all scan classes.

“Scheduled Scans: Scan count this interval” (.sched_scans_this_interval)

A *.sched_scans_this*_interval performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_this_interval performance counter point indicates the number of scans that the interface performed per performance summary interval for the scan class or the total number of scans performed for all scan classes during the summary interval. This point is similar to the [UI_SCSCANCOUNT] health point.

The ICU uses a naming convention such that the point containing “(Scan Class 1)” (for example, "`sy.perf.etamp390.E1(Scan Class 1).sched_scans_this_interval`") refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing “(_Total)” refers to the sum of all scan classes.

Performance Counters for (_Total) only

“Device Actual Connections” (.Device_Actual_Connections)

The *.Device_Actual_Connections* performance counter point stores the actual number of foreign devices currently connected and working properly out of the expected number of foreign device connections to the interface. This value will always be less than or equal to the Device Expected Connections counter.

“Device Expected Connections” (.Device_Expected_Connections)

The *.Device_Expected_Connections* performance counter point stores the total number of foreign device connections for the interface. This is the expected number of foreign device connections configured that should be working properly at runtime. If the interface can only communicate with 1 foreign device then the value of this counter will always be one. If the interface can support multiple foreign device connections then this is the total number of expected working connections configured for this interface.

“Device Status” (.Device_Status)

The *.Device_Status* performance counter point stores communication information about the interface and the connection to the foreign device(s). The value of this counter is based on the expected connections, actual connections and value of the **`/PercentUp`** command line option. If the device status is good then the value is ‘0’. If the device status is bad then the value is ‘1’. If the interface only supports connecting to 1 foreign device then the **`/PercentUp`** command line value does not change the results of the calculation. If for example the interface can connect to 10 devices and 5 are currently working then the value of the **`/PercentUp`** command line parameter is applied to determine the Device Status. If the value of the **`/PercentUp`** command line parameter is set to 50 and at least 5 devices are working then the DeviceStatus will remain good (that is, have a value of zero).

"Failover Status" (.Failover_Status)

The *.Failover_Status* performance counter point stores the failover state of the interface when configured for UniInt failover. The value of the counter will be '0' when the interface is running as the primary interface in the failover configuration. If the interface is running in backup mode then the value of the counter will be '1'.

"Interface up-time (seconds)" (.up_time)

The *.up_time* performance counter point indicates the amount of time (in seconds) that this interface has been running. At startup the value of the counter is zero. The value will continue to increment until it reaches the maximum value for an unsigned integer. Once it reaches this value then it will start back over at zero.

"IO Rate (events/second)" (.io_rates)

The *.io_rates* performance counter point indicates the rate (in event per second) at which this interface writes data to its input points.

"Log file message count" (.log_file_msg_count)

The *.log_file_msg_count* performance counter point indicates the number of messages that the interface has written to the log file. This point is similar to the [UI_MSGCOUNT] health point.

"PI Status" (PI_Status)

The *.PI_Status* performance counter point stores communication information about the interface and the connection to the PI Data Archive. If the interface is properly communicating with the PI Data Archive then the value of the counter is '0'. If the communication to the PI Data Archive goes down for any reason then the value of the counter will be '1'. Once the interface is properly communicating with the PI Data Archive again then the value will change back to '0'.

"Points added to the interface" (.pts_added_to_interface)

The *.pts_added_to_interface* performance counter point indicates the number of points the interface has added to its point list. This does not include the number of points configured at startup. This is the number of points added to the interface after the interface has finished a successful startup.

"Points edited in the interface"(.pts_edited_in_interface)

The *.pts_edited_in_interface* performance counter point indicates the number of point edits the interface has detected. The interface detects edits for those points whose PointSource attribute matches the **`/ps=`** parameter and whose Location1 attribute matches the **`/id=`** parameter of the interface.

"Points Good" (.Points_Good)

The *.Points_Good* performance counter point is the number of points that have sent a good current value to the PI Data Archive. A good value is defined as any value that is not a system digital state value. A point can either be Good, In Error, or Stale. The total of Points

Good, Points In Error, and Points State will equal the Point Count. There is one exception to this rule. At startup of an interface, the Stale timeout must elapse before the point will be added to the Stale Counter. Therefore the interface must be up and running for at least 10 minutes for all points to belong to a particular Counter.

"Points In Error" (.Points_In_Error)

The *.Points_In_Error* performance counter point indicates the number of points that have sent a current value to the PI Data Archive that is a system digital state value. Once a point is in the In Error count it will remain in the In Error count until the point receives a new, good value. Points categorized as in Error do not transition to the Stale Counter. Only good points become stale.

"Points removed from the interface" (.pts_removed_from_interface)

The *.pts_removed_from_interface* performance counter point indicates the number of points that have been removed from the interface configuration. A point can be removed from the interface when one of the point attributes is updated and the point is no longer a part of the interface configuration. For example, changing the PointSource, Location1, or Scan attribute can cause the point to no longer be a part of the interface configuration.

"Points Stale 10(min)" (.Points_Stale_10min)

The *.Points_Stale_10min* performance counter point indicates the number of good points that have not received a new value in the last 10 minutes. If a point is Good, then it will remain in the good list until the Stale timeout elapses. At this time if the point has not received a new value within the Stale Period then the point will move from the Good count to the Stale count. Only points that are Good can become Stale. If the point is in the In Error count then it will remain in the In Error count until the error clears. As stated above, the total count of Points Good, Points In Error, and Points Stale will match the Point Count for the interface.

"Points Stale 30(min)" (.Points_Stale_30min)

The *.Points_Stale_30min* performance counter point indicates the number of points that have not received a new value in the last 30 minutes. For a point to be in the Stale 30 minute count it must also be a part of the Stale 10 minute count.

"Points Stale 60(min)" (.Points_Stale_60min)

The *.Points_Stale_60min* performance counter point indicates the number of points that have not received a new value in the last 60 minutes. For a point to be in the Stale 60 minute count it must also be a part of the Stale 10 minute and 30 minute count.

"Points Stale 240(min)" (.Points_Stale_240min)

The *.Points_Stale_240min* performance counter point indicates the number of points that have not received a new value in the last 240 minutes. For a point to be in the Stale 240 minute count it must also be a part of the Stale 10 minute, 30 minute and 60 minute count.

Performance Counters for (Scan Class x) only

"Device Scan Time (milliseconds)" (.Device_Scan_Time)

A *.Device_Scan_Time* performance counter point is available for each scan class of this interface.

The *.Device_Scan_Time* performance counter point indicates the number of milliseconds the interface takes to read the data from the foreign device and package the data to send to the PI Data Archive. This counter does not include the amount of time to send the data to the Pi Data Archive. This point is similar to the [UI_SCINDEVSCANTIME] health point.

The ICU uses a naming convention such that the point containing "(Scan Class 1)" (for example, "`sy.perf.etamp390.E1 (Scan Class 1).device_scan _time`") refers to scan class 1, "(Scan Class 2) refers to scan class 2, and so on.

"Scan Time (milliseconds)" (.scan_time)

A *.scan_time* performance counter point is available for each scan class of this interface.

The *.scan_time* performance counter point indicates the number of milliseconds the interface takes to both read the data from the device and send the data to the PI Data Archive. This point is similar to the [UI_SCINSCANTIME] health point.

The ICU uses a naming convention such that the point containing "(Scan Class 1)" (for example, "`sy.perf.etamp390.E1(Scan Class 1).scan_time"`) refers to scan class 1, "(Scan Class 2)" refers to scan class 2, and so on.

Interface Health Monitoring Points

Interface Health Monitoring Points provide information about the health of this interface. To use the ICU to configure these points, select this interface from the *Interface* drop-down list and click *Health Points* from the parameter category pane:

Right click the row for a particular health point to display the shortcut menu:

Click *Create* to create the health point for that particular row. Click *Create All* to create all the health points.

To see the current values (snapshots) of the health points, right-click and select *Refresh Snapshots*.

For some of the health points described subsequently, the interface updates their values at each performance summary interval (typically, 8 hours).

[UI_HEARTBEAT]

The [UI_HEARTBEAT] health point indicates whether the interface is currently running. The value of this point is an integer that increments continuously from 1 to 15. After reaching 15, the value resets to 1.

The fastest scan class frequency determines the frequency at which the interface updates this point:

Fastest Scan Frequency	Update frequency
Less than 1 second	1 second
Between 1 and 60 seconds, inclusive	Scan frequency
More than 60 seconds	60 seconds

If the value of the [UI_HEARTBEAT] health point is not changing, then this interface is in an unresponsive state.

[UI_DEVSTAT]

The [UI_DEVSTAT] health point provides an indication of the connection status between the interface and the source device. The possible values for this string point are:

- "1 | Starting | UI #.#.#.#" – The interface is currently starting up.
- "Good" – The interface is properly communicating and reading data from all configured PMUs. A value of "Good" does not mean that all points are receiving good values, but it is a good indication that there are no hardware or network problems.
- "3 | 1 devices(s) in error | REP: 1 LEP 1" – The connection to one or more PMUs has failed.
- "4 | Intf Shutdown" – The interface has been shutdown.

Refer to the *PI Universal Interface (UniInt) User Guide*.doc file for more information on how to configure health points.

[UI_SCINFO]

The [UI_SCINFO] health point provides scan class information. The value of this point is a string that indicates

- the number of scan classes;
- the update frequency of the [UI_HEARTBEAT] health point ; and
- the scan class frequencies

An example value for the [UI_SCINFO] health point is:

```
3 | 5 | 5 | 60 | 120
```

The interface updates the value of this point at startup and at each performance summary interval.

[UI_IORATE]

The [UI_IORATE] health point indicates the sum of

1. the number of scan-based input values the interface collects before it performs exception reporting; and
2. the number of event-based input values the interface collects before it performs exception reporting

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The value of this [UI_IORATE] health point may be zero. A stale timestamp for this point indicates that this interface has stopped collecting data.

[UI_MSGCOUNT]

The [UI_MSGCOUNT] health point tracks the number of messages that the interface has written to the log file since start-up. In general, a large number for this point indicates that the interface is encountering problems. You should investigate the cause of these problems by looking in log messages.

The interface updates the value of this point every 60 seconds. While the interface is running, the value of this point never decreases.

[UI_POINTCOUNT]

The [UI_POINTCOUNT] health point counts number of PI points loaded by the interface. This count includes all input and triggered input points. This count does NOT include any interface health points or performance points.

The interface updates the value of this point at startup, on change and at shutdown.

[UI_TRIGGERRATE]

The [UI_TRIGGERRATE] health point tracks the number of values that the interface writes to event-based input points. If there are no event-based input points for this interface, it writes the System Digital State `No Result` to this health point.

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The interface resets the value of this point to zero at each performance summary interval.

[UI_TRIGGERBVRATE]

The [UI_TRIGGERBVRATE] health point tracks the number of system digital state values that the interface writes to event-based input points. If there are no event-based input points for this interface, it writes the system digital state `No Result` to this health point.

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The interface resets the value of this point to zero at each performance summary interval.

[UI_SCIORATE]

You can create a [UI_SCIORATE] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class IO Rate.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCIORATE] health point indicates the number of values that the interface has collected. If the current value of this point is between zero and the corresponding [UI_SCPOINTCOUNT] point, inclusive, then the interface executed the scan successfully. If a [UI_SCIORATE] point stops updating, then this condition indicates that an error has occurred and the points in the scan class are no longer receiving new data.

The interface updates the value of a [UI_SCIORATE] point after the completion of the associated scan.

Although the ICU allows you to create the point with the suffix ".sc0", this point is not applicable to this interface.

[UI_SCBVRATE]

You can create a [UI_SCBVRATE] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Bad Value Rate.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCBVRATE] point indicates the number System Digital State values that the interface has collected.

The interface updates the value of a [UI_SCBVRATE] point after the completion of the associated scan.

Although the ICU allows you to create the point with the suffix ".sc0", this point is not applicable to this interface.

[UI_SCSCANCOUNT]

You can create a [UI_SCSCANCOUNT] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Scan Count.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_ SCSCANCOUNT] point tracks the number of scans that the interface has performed.

The interface updates the value of this point at the completion of the associated scan. The interface resets the value to zero at each performance summary interval.

Although there is no "Scan Class 0", the ICU allows you to create the point with the suffix ".sc0". This point indicates the total number of scans the interface has performed for all of its Scan Classes.

[UI_SCSKIPPED]

You can create a [UI_SCSKIPPED] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Scans Skipped.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCSKIPPED] point tracks the number of scans that the interface was not able to perform before the scan time elapsed and before the interface performed the next scheduled scan.

The interface updates the value of this point each time it skips a scan. The value represents the total number of skipped scans since the previous performance summary interval. The interface resets the value of this point to zero at each performance summary interval.

Although there is no “Scan Class 0”, the ICU allows you to create the point with the suffix “.sc0”. This point monitors the total skipped scans for all of the interface’s Scan Classes.

[UI_SCPOINTCOUNT]

You can create a [UI_SCPOINTCOUNT] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Point Count.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

This health point monitors the number of points in a scan class.

The interface updates a [UI_SCPOINTCOUNT] health point when it performs the associated scan.

Although the ICU allows you to create the point with the suffix “.sc0”, this point is not applicable to this interface.

[UI_SCINSCANTIME]

You can create a [UI_SCINSCANTIME] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Scan Time.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

A particular scan class’’ [UI_ SCINSCANTIME] point represents the amount of time (in milliseconds) the interface takes to read data from the device, fill in the values for the points, and send the values to the PI Data Archive.

The interface updates the value of this point at the completion of the associated scan.

[UI_SCINDEVSCANTIME]

You can create a [UI_SCINDEVSCANTIME] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Device Scan Time.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

A particular scan class'' [UI_ SCINDEVSCANTIME] point represents the amount of time (in milliseconds) the interface takes to read data from the device and fill in the values for the points.

The value of a [UI_ SCINDEVSCANTIME] point is a fraction of the corresponding [UI_SCINSCANTIME] point value. You can use these numbers to determine the percentage of time the interface spends communicating with the device compared with the percentage of time communicating with the PI Data Archive.

If the [UI_SCSKIPPED] value is increasing, the [UI_SCINDEVSCANTIME] points along with the [UI_SCINSCANTIME] points can help identify where the delay is occurring: whether the reason is communication with the device, communication with the PI Data Archive, or elsewhere.

The interface updates the value of this point at the completion of the associated scan.

I/O Rate Point

An I/O Rate point measures the rate at which the interface writes data to its input points. The value of an I/O Rate point represents a 10-minute average of the total number of values per minute that the interface sends to the PI Data Archive.

When the interface starts, it writes 0 to the I/O Rate point. After running for ten minutes, the interface writes the I/O Rate value. The interface continues to write a value every 10 minutes. When the interface stops, it writes 0.

The ICU allows you to create one I/O Rate point for each copy of this interface. Select this interface from the *Interface* drop-down list, click *IO Rate* in the parameter category pane, and check *Enable IORates for this interface*.

As the preceding picture shows, the ICU suggests an *Event Counter* number and a *Tagname* for the I/O Rate point. Click the *Save* button to save the settings and create the I/O Rate point. Click the *Apply* button to apply the changes to this copy of the interface.

You need to restart the interface in order for it to write a value to the newly created I/O Rate point. Restart the interface by clicking the *Restart* button:

(The reason you need to restart the interface is that the PointSource attribute of an I/O Rate point is `Lab`.)

To confirm that the interface recognizes the I/O Rate point, look in the log file for a message such as:

```
PI-ModBus 1> IORATE: tag sy.io.etamp390.ModbusE1 configured.
```

To see the I/O Rate point's current value (snapshot), click the *Refresh* button of the *Snapshot* box:

Enable IORates for this Interface

The *Enable IORates for this interface* check box enables or disables I/O Rates for the current interface. To disable I/O Rates for the selected interface, uncheck this box. To enable I/O Rates for the selected interface, check this box.

Event Counter

The *Event Counter* correlates a point specified in the iorates.dat file with this copy of the interface. The command-line equivalent is **`/ec=`***x*, where *x* is the same number that is assigned to a point name in the iorates.dat file.

Tagname

The name listed in the *Tagname* box is the name of the I/O Rate point.

Tag Status

The *Tag Status* box indicates whether the I/O Rate point exists in the PI Data Archive. The possible states are:

- Created – This status indicates that the point exist in the PI Data Archive
- Not Created – This status indicates that the point does not yet exist in the PI Data Archive
- Deleted – This status indicates that the point has just been deleted
- Unknown – This status indicates that the PI ICU is not able to access the PI Data Archive

In File

The *In File* box indicates whether the I/O Rate point listed in the *Tagname* box and the number in the *Event Counter* is in the IORates.dat file. The possible states are:

- Yes – This status indicates that the point name and event counter are in the IORates.dat file
- No – This status indicates that the point name and event counter are not in the IORates.dat file

Snapshot

The *Snapshot* column holds the snapshot value of the I/O Rate point, if the I/O Rate point exists in the PI Data Archive. The *Snapshot* box is updated when the *IORate* page is selected, and when the interface is first loaded.

Create/Save

Create the suggested I/O Rate point with the name indicated in the Tagname box. Or, save any changes for the point name indicated in the Tagname box.

Delete

Delete the I/O Rate point listed in the *Tagname* box.

Rename

Change the name for the I/O Rate point listed in the *Tagname* box.

Add to File

Add the point indicated by the name in the Tagname box to the IORates.dat file with the event counter listed in the *Event Counter* box.

Search

Search the PI Data Archive for a previously defined I/O Rate point.

Appendix A. Error and Informational Messages

A string `NameID` is pre-pended to error messages written to the message log. `Name` is a non-configurable identifier that is no longer than 9 characters. *ID* is a configurable identifier that is no longer than 9 characters and is specified using the **`/id`** parameter on the startup command-line.

Message Logs

The location of the message log depends upon the platform on which the interface is running. See the *PI Universal Interface (UniInt) User Guide* for more information.

For more information about logs for interfaces running on Windows, see *UniInt Interface Message Logging for UniInt 4.5.0.x and later Interfaces* or OISsoft KB article - KB00401 - How to read new UniInt 4.5.0.x and later Interface message logs on the OSIsoft technical support web site.

Messages are written to `[PIHOME]\dat\pipc.log` at the following times.

- When the interface starts many informational messages are written to the log. These include the version of the interface, the version of UniInt, the command-line parameters used, and the number of points.
- As the interface loads points, messages are sent to the log if there are any problems with the configuration of the points.
- If the UniInt **`/dbUniInt`** parameter is found in the command-line, then various informational messages are written to the log file.

Messages

Interface Configuration Error Messages

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Schema instance, make sure MSXML 4.0 is installed.`
	Or
	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Document instance, make sure MSXML 4.0 is installed.`
	Or
	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Schema Collection object, make sure MSXML 4.0 is installed.`

Cause	There was a problem created one of the DOM XML objects. Typically indicates that the Microsoft MSXML 4.0 API is not installed.
Resolution	Download the API from Microsoft and install.

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig – XML config document load failed: System error: -2146697210.`
Cause	The specified XML file could not be loaded.
Resolution	Check the file name and path specified via the /confile argument.

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't find root <PI_C37_118.Session> element in the XML configuration document.`
Cause	The root node root <PI_C37_118.Session> couldn't be located.
Resolution	Check the format of the XML file. If using the PIC37118 ICU (recommended) this should be correct.

Message	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for DataSortingAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for SyncErrorAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for PMUErrorActionmust be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for DataSortingAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for LeapSecondAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for WriteIOTimeout must be 0 or 1.`

Cause	The value specified for the one of the above attributes in the root section of the XML configuration file is out of range.
Resolution	Use PIC37118 ICU to ensure this value is correct.

Message	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - No <LocalEndPoint> elements found in the PIC37118_Arbiter_Example.xml XML configuration document.`
Cause	No <LocalEndPoints> section was found in the configuration document.
Resolution	The must be at least one LocalEndPoints section in the file.
Message	`LocalEndPoint::LocalEndPoint( -1 ) - ID attribute missing from LocalEndPoint definition.`
Cause	The required ID attribute is missing from a LocalEndPoint section.
Resolution	Add the ID attribute to the LocalEndPoint definition.

Message	`LocalEndPoint::LocalEndPoint( 2 ) - Type element in LocalEndpoint configuration must be IP or Serial.`
Cause	The Type attribute in LocalEndPoint with ID of 2 has invalid communications type specified.
Resolution	Use the PIC37118 ICU to ensure this attribute is correct.

Message	`LocalEndPoint::LocalEndPoint( 2 ) - Invalid data protocol specified in <DataProtocol>, must be TCP or UDP(case insensitive).`
	Or
	`LocalEndPoint::LocalEndPoint( 2 ) - Invalid data protocol specified in <ConfigProtocol>, must be TCP or UDP(case insensitive).`

Cause	The DataProtocol or ConfigProtocol attribute must be TCP or UDP.
Resolution	Use the PIC37118 ICU to ensure this attribute is correct. This parameter is not case sensitive.

Point Loading Error Messages

The following messages are logged when there is a problem loading a point. In each case the point name and point id is reported along with the error information.

Message	`LocalEndPoint Id: 1 not found in interface LocalEndPointCollection`
Cause	There is not a LocalEndPoint defined in the XML configuration file that matches the Location2 attribute of the point.
Resolution	Add the appropriate LocalEndPoint.

Message	`RemoteEndPoint Id: 1 not found in interface RemoteEndPointCollection`
Cause	There is not a RemoteEndPoint defined in the XML configuration file that matches the Location3 attribute of the point.
Resolution	Add the appropriate RemoteEndPoint

Message	`Delimiter character '\' not found in InstrumentTag specification (PMU.FREQ)`
Cause	The InstrumentTag attribute requires the Measurement Type and Measurement be delimited by a "/" character and was not found.
Resolution	Added appropriate delimiter.
Message	`RemoteEndPoint tag of type LEAPSEC is already defined`
Cause	There can only be a single point for a given RemoteEndPoint type(REP) for each RemoteEndPoint. For example there cannot be two points with an InstrumentTag field of REP/LEAPSEC configured for the same RemoteEndPoint. Any subsequent points are rejected.
Resolution	Remove duplicate point.

Message	`PMU Id: 66 not found in interface PMU Collection`
Cause	The PMU ID specified in the Location5 attribute was not found in the XML configuration file.
Resolution	Fix XML configuration file to include ID.

Message	`Phasor measurement string XX is shorter then shortest valid keyword`
	Or
	`Invalid Phasor sub type ( XXX ) specified`

Cause	The Phasor measurement specified in the InstrumentTag attribute is invalid. This value must be PhasorReal, PhasorImag, PhasorMag or PhasorAngle.
Resolution	Correct XML configuration file.

Message	`Phasor channel ( XXX ) is not configured`
	`Or`
	`Analog channel ( XXX ) is not configured`
	`Or`
	`Digital channel ( XXX ) is not configured`

Cause	The Phasor, analog or digital channel name is not valid for the PMU. This value is not case sensitive but imbedded spaces must remain if they are present in the channel names reported by the PMU. Trailing spaces may be removed.
Resolution	Correct XML configuration file.

Message	`Phasor tag with channel name of XXX and sub-type of IMAG is already defined for PMU`
	Or
	`Analog tag with channel name of XXX is already defined for PMU`
	Or
	`Digital tag with channel name of XXX is already defined for PMU`

Cause	There can only be one PI point configured for each Analog or Digital channel. Each Phasor channel can have one PI point for each sub type on a channel.
Resolution	Correct by removing the duplicate point entry.

Message	`Invalid PMU measurement type ( XXX ) specified`
Cause	The PMU measurement specified in the InstrumentTag is not valid. For example: PMU/XXX is not valid but PMU/FREQ is.
Resolution	Correct the invalid measurement type.

Communications Related Error Messages

The following messages are logged when there is a problem communications to the C37.118 device. In each case the communications parameters are reported along with the error information.

Message	`PDC/PMU communication timeout.`
Cause	The communications with a PDC or PMU have timed out. The interface will continue to attempt a reconnection.

Message	`Error code: -1, Generic Error`
Cause	The interface was unable to determine the exact nature of the error. Any available information is displayed.

Message	`Debug code: -2, Debug Message.`
Cause	Miscellaneous debug message.

Message	`Error code: -3, Truncated String.`
Cause	The C37.118 message was truncated.

Message	`Error code: -4, Failed to create communication instance.`
Cause	The interface was unable to create a communications thread. Check system resources.
Resolution	Check system resources.

Message	`Error code: -5, Bad memory allocation.`
Cause	The UFO_ID pa
Resolution	Check system resources.

Message	`Error code: -6, Required parameter passed a NULL pointer.`
Cause	An internal function passed a NULL pointer for a required variable. This should never occur.
Resolution	Contact OSIsoft technical support.

Message	`Error code: -7, Failed to stop communication thread.`
Cause	The communications thread could not be stopped. This would only occur during interface shutdown, but should never occur.
Resolution	Contact OSIsoft technical support.

Message	`Error code: -8, PDC/PMU communication timeout.`
Cause	The communications with a PDC or PMU have timed out. The interface will continue to attempt a reconnection.

Message	`Error code: -9, Sync byte not found.`
Cause	The 0xAA sync byte was not found in the data stream. The interface will attempt to resynchronize the data stream by closing the socket and reconnecting.
Message	`Error code: -10, Message IDCODE not for PDC / PMU.`
Cause	The message received was valid but the IDCODE in word 3 of the frame was not for the configured PDC / PMU.
Resolution	Check the XML configuration file and PDC / PMU configuration.

Message	`Error code: -11, Message frame size error.`
Cause	The frame size in the C37.118 message did not match the actual size.

Message	`Error code: -12, Message CRC error.`
Cause	The CRC validation failed on a C37.118 message.
Resolution	This error may occur occasionally. If it occurs with any frequency, check the communications hardware.

Message	`Error code: -13, No Message Buffer Available.`
Cause	No message buffer was available. The interface will continue to operate but will be missing data.
Resolution	Contact OSIsoft technical support.

System Errors and PI Errors

System errors are associated with positive error numbers. Errors related to PI are associated with negative error numbers.

Error Descriptions

Descriptions of system and PI errors can be obtained with the pidiag utility:

```
\PI\adm\pidiag /e error_number
```

UniInt Failover Specific Error Messages

Informational

Message	`16-May-06 10:38:00` `C37118 1> UniInt failover: Interface in the “Backup” state.`
Meaning	Upon system startup, the initial transition is made to this state. While in this state the interface monitors the status of the other interface participating in failover. When configured for Hot failover, data received from the data source is queued and not sent to the PI Data Archive while in this state. The amount of data queued while in this state is determined by the failover update interval. In any case, there will be typically no more than two update intervals of data in the queue at any given time. Some transition chains may cause the queue to hold up to five failover update intervals worth of data.

Message	`16-May-06 10:38:05` `C37118 1> UniInt failover: Interface in the “Primary” state and actively sending data to PI. Backup interface not available.`
Meaning	While in this state, the interface is in its primary role and sends data to the PI Data Archive as it is received. This message also states that there is not a backup interface participating in failover.

Message	`16-May-06 16:37:21` `C37118 1> UniInt failover: Interface in the “Primary” state and actively sending data to PI. Backup interface available.`
Meaning	While in this state, the interface sends data to the PI Data Archive as it is received. This message also states that the other copy of the interface appears to be ready to take over the role of primary.

Errors (Phase 1 & 2)

Message	`16-May-06 17:29:06`
	`C37118 1> One of the required Failover Synchronization points was not loaded.`
	`Error = 0: The Active ID synchronization point was not loaded.`
	`The input PI tag was not loaded`

Cause	The active ID point is not configured properly.
Resolution	Check validity of point attributes. For example, make sure Location1 attribute is valid for the interface. All failover tags must have the same `PointSource` and `Location1` attributes. Modify point attributes as necessary and restart the interface.

Message	`16-May-06 17:38:06`
	`C37118 1> One of the required Failover Synchronization points was not loaded.`
	`Error = 0: The Heartbeat point for this copy of the interface was not loaded.`
	`The input PI tag was not loaded`

Cause	The heartbeat point is not configured properly.
Resolution	Check validity of point attributes. For example, make sure Location1 attribute is valid for the interface. All failover tags must have the same PointSource and Location1 attributes. Modify point attributes as necessary and restart the interface.

Message	`17-May-06 09:06:03` `C37118 1> The Uniint FailOver ID (`**`/UFO_ID`**`) must be a positive integer.`
Cause	The UFO_ID parameter has not been assigned a positive integer value.
Resolution	Change and verify the parameter to a positive integer and restart the interface.

Message	`17-May-06 09:06:03` `C37118 1> The Failover ID parameter (`**`/UFO_ID`**`) was found but the ID for the redundant copy was not found`
Cause	The **`/UFO_OtherID`** parameter is not defined or has not been assigned a positive integer value.
Resolution	Change and verify the **`/UFO_OtherID`** parameter to a positive integer and restart the interface.

Errors (Phase 2)

Unable to open synchronization file

Message	27-Jun-08 17:27:17 PI Eight Track 1 1> Error 5: Unable to create file '\\georgiaking\GeorgiaKingStorage\UnIntFailover\\PIEightTrack_eight_1.dat' Verify that interface has read/write/create access on file server machine. Initializing UniInt library failed Stopping Interface
Cause	This message will be seen when the interface is unable to create a new failover synchronization file at startup. The creation of the file only takes place the first time either copy of the interface is started and the file does not exist. The error number most commonly seen is error number 5. Error number 5 is an “access denied” error and is likely the result of a permissions problem.
Resolution	Ensure the account the interface is running under has read and write permissions for the folder. The “log on as” property of the Windows service may need to be set to an account that has permissions for the folder.

Error Opening Synchronization File

Message	Sun Jun 29 17:18:51 2008 PI Eight Track 1 2> WARNING> Failover Warning: Error = 64 Unable to open Failover Control File '\\georgiaking\GeorgiaKingStorage\Eight\PIEightTrack_eight_1.dat' The interface will not be able to change state if PI is not available
Cause	This message will be seen when the interface is unable to open the failover synchronization file. The interface failover will continue to operate correctly as long as communication to the PI Data Archive is not interrupted. If communication to the PI Data Archive is interrupted while one or both interfaces cannot access the synchronization file, the interfaces will remain in the state they were in at the time of the second failure, so the primary interface will remain primary and the backup interface will remain backup.
Resolution	Ensure the account the interface is running under has read and write permissions for the folder and file. The “log on as” property of the Windows service may need to be set to an account that has permissions for the folder and file.

Appendix B. PI SDK Options

To access the PI SDK settings for this interface, select this interface from the *Interface* drop-down list and click *UniInt – PI SDK* in the parameter category pane.

Disable PI SDK

Select *Disable PI SDK* to tell the interface not to use the PI SDK. If you want to run the interface in disconnected startup mode, you must choose this option.

The command line equivalent for this option is **`/pisdk=0`**.

Use the Interface's default setting

This selection has no effect on whether the interface uses the PI SDK. However, you must not choose this option if you want to run the interface in disconnected startup mode.

Enable PI SDK

Select *Enable PI SDK* to tell the interface to use the PI SDK. Choose this option if the PI Data Archive version is earlier than 3.4.370.x or the PI API is earlier than 1.6.0.2, and you want to use extended lengths for the Tag, Descriptor, ExDesc, InstrumentTag, or PointSource point attributes. The maximum lengths for these attributes are:

Attribute	Enable the Interface to use the PI SDK	PI Data Archive earlier than 3.4.370.x or PI API earlier than 1.6.0.2, without the use of the PI SDK
Tag	1023	255
Descriptor	1023	26
ExDesc	1023	80
InstrumentTag	1023	32
PointSource	1023	1

However, if you want to run the interface in disconnected startup mode, you must not choose this option.

The command line equivalent for this option is **`/pisdk=1`**.

Appendix C. Communication Plug-In Selection

This appendix describes the selection criteria for the Communications plug-in for the PDCs and PMUs which have been tested. Current Plug-In selections are PIC37118_Comm001, PIC37118_Comm002 and PIC37118_Comm003.

PDC/PMU	Data Protocol	Command Protocol	Command and Data on Same Port	Data Transmission Mode	Close Cmd Comm Setting	Plug-In
Arbiter 1133	UDP	UDP	Yes	Unicast	Yes	001
Arbiter 1133	TCP	TCP	Yes	Unicast	Yes	001
Arbiter 1133	TCP	UDP	No	Unicast	Yes	002
Arbiter 1133	TCP	UDP	No	Broadcast	Yes	002
Arbiter 1133	TCP	UDP	No	Multicast	Yes	002
Areva 897	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
Electric Power Group PDC	UDP	TCP	No	Unicast	Yes	002
NI Rio	TCP	TCP	Yes	Unicast	Yes	001
Siemens R-PMU	TCP	TCP	Yes	Unicast	Yes	001
SEL 3306	TCP	UDP	No	Unicast	No	002
SEL 3378 (SVP operating in PDC mode)	TCP	TCP	Yes	Unicast	Yes	001
SEL 421	UDP	UDP	Yes	Unicast	Yes	001
SEL 421	TCP	TCP	Yes	Unicast	Yes	001
SEL 421	TCP	UDP	No	Unicast	Yes	002
SEL 421	TCP	UDP	No	Broadcast	Yes	002
SEL 421	TCP	UDP	No	Multicast	Yes	002
SEL 451	UDP	UDP	Yes	Unicast	Yes	001
SEL 451	TCP	TCP	Yes	Unicast	Yes	001
SEL 451	TCP	UDP	No	Unicast	Yes	002
SEL 451	TCP	UDP	No	Broadcast	Yes	002
SEL 451	TCP	UDP	No	Multicast	Yes	002
SEL 487E	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
Siemens R-PMU	TCP	TCP	Yes	Unicast	Yes	001
TVA Proxy	Udp	n/a	n/a	Unicast	No	003
Serial (All Devices)	n/a	n/a	n/a	n/a	No	001

Appendix D. Terminology

To understand this interface manual, you should be familiar with the terminology used in this document.

Interface Specific Terms

IEEE

IEEE stands for the Institute of Electrical & Electronic Engineers.

PMU

A Phasor Measurement Unit (PMU) is a generic device that produces synchronized phasors from voltage and/or current inputs and synchronizing signal.

Phasor

A Phasor is the complex equivalent of a simple sine wave quantity such that the complex modulus is the sine wave amplitude and complex angle (in polar form) is the sine wave phase angle.

PDC

A Phasor Data Concentrator (PDC) provides Phasor data from multiple PMUs.

General Terms

Buffering

Buffering refers to an interface node's ability to store temporarily the data that interfaces collect and to forward these data to the appropriate PI Data Archives.

N-Way Buffering

If you have PI Data Archives that are part of a collective, PIBufss supports n-way buffering. N-way buffering refers to the ability of a buffering application to send the same data to each of the PI Data Archives in a collective. (Bufserv also supports n-way buffering to multiple PI Data Archives however it does not guarantee identical archive records since point compressions attributes could be different between PI Data Archives. With this in mind, OSIsoft recommends that you run PIBufss instead.)

ICU

ICU refers to the PI Interface Configuration Utility. The ICU is the primary application that you use to configure PI interface programs. You must install the ICU on the same computer on which an interface runs. A single copy of the ICU manages all of the interfaces on a particular computer.

You can configure an interface by editing a startup command file. However, OSIsoft discourages this approach. Instead, OSIsoft strongly recommends that you use the ICU for interface management tasks.

ICU Control

An ICU Control is a plug-in to the ICU. Whereas the ICU handles functionality common to all interfaces, an ICU Control implements interface-specific behavior. Most PI interfaces have an associated ICU Control.

Interface Node

An interface node is a computer on which

- the PI API and/or PI SDK are installed, and
- PI Data Archive programs are not installed.

PI API

The PI API is a library of functions that allow applications to communicate and exchange data with the PI Data Archive. All PI interfaces use the PI API.

PI Data Archive Collective

A PI Data Archive Collective is two or more replicated PI Data Archives that collect data concurrently. Collectives are part of the High Availability environment. When the primary PI Data Archive in a collective becomes unavailable, a secondary collective member node seamlessly continues to collect and provide data access to your PI clients.

PI Data Archive Node

A PI Data Archive node is a computer on which PI Data Archive programs are installed. The PI Data Archive runs on the PI Data Archive node. In earlier documentation, PI Data Archive was referred to as the PI Server. (See PI Server Node.)

PIHOME

PIHOME refers to the directory that is the common location for PI 32-bit client applications.

A typical *PIHOME* on a 32-bit operating system is `C:\Program Files\PIPC`.

A typical *PIHOME* on a 64-bit operating system is `C:\Program Files (x86)\PIPC`.

PI 32-bit interfaces reside in a subdirectory of the `Interfaces` directory under `PIHOME`.

For example, files for the 32-bit Modbus Ethernet Interface are in

`[PIHOME]\PIPC\Interfaces\ModbusE`.

This document uses `[PIHOME]` as an abbreviation for the complete `PIHOME` or `PIHOME64` directory path. For example, ICU files in `[PIHOME]\ICU`.

PIHOME64

PIHOME64 is found only on a 64-bit operating system and refers to the directory that is the common location for PI 64-bit client applications.

A typical *PIHOME64* is `C:\Program Files\PIPC`.

PI 64-bit interfaces reside in a subdirectory of the `Interfaces` directory under *PIHOME64*.

For example, files for a 64-bit Modbus Ethernet Interface would be found in

```
C:\Program Files\PIPC\Interfaces\ModbusE.
```

This document uses `[PIHOME]` as an abbreviation for the complete `PIHOME` or *PIHOME64* directory path. For example, ICU files in `[PIHOME]\ICU`.

PI Message Log

The PI message log is the file to which OSIsoft interfaces based on UniInt 4.5.0.x and later write informational, debug and error messages. When a PI interface runs, it writes to the local PI message log. This message file can only be viewed using the PIGetMsg utility. See the *UniInt Interface Message Logging.docx* file for more information on how to access these messages.

PI SDK

The PI SDK is a library of functions that allow applications to communicate and exchange data with the PI Data Archive. Some PI interfaces, in addition to using the PI API, require the use of the PI SDK.

PI Server Node

In earlier documentation, the term "PI Server" was used as a nickname for the PI Data Archive and a PI Server node was a computer on which PI Data Archive programs were installed. While the PI Data Archive remains a core server of the PI Server product, the product name "PI Server" now refers to much more than the PI Data Archive. OSIsoft documentation, including this user manual, is changing to use "PI Server" in this broader sense and "PI Data Archive" to refer to the historian core. (See PI Data Archive Node.)

PI SMT

PI SMT refers to PI System Management Tools. PI SMT is the program that you use for configuring PI Data Archives. A single copy of PI SMT manages multiple PI Data Archives. PI SMT runs on either a PI Data Archive node or an interface node.

Pipc.log

The `pipc.log` file is the file to which OSIsoft interfaces based on UniInt versions earlier than 4.5.0.x write informational and error messages. When a PI interface runs, it writes to the `pipc.log` file. The ICU allows easy access to the `pipc.log`.

Point

The PI point is the basic building block for controlling data flow to and from the PI Data Archive. For a given timestamp, a PI point holds a single value.

A PI point does not necessarily correspond to a "point" on the data source device. For example, a single "point" on the data source device can consist of a set point, a process value,

an alarm limit, and a discrete value. These four pieces of information require four separate PI points.

Service

A Service is a Windows program that runs without user interaction. A service continues to run after you have logged off from Windows. It has the ability to start up when the computer itself starts up.

The ICU allows you to configure a PI interface to run as a service.

Tag

The Tag attribute of a PI point is the name of the PI point. There is a one-to-one correspondence between the name of a point and the point itself. Because of this relationship, PI System documentation uses the terms “tag” and “point” interchangeably.

Interfaces read values from a device and write these values to an input tag.

Appendix E. Technical Support and Resources

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or techsupport@osisoft.com. The OSIsoft Technical Support website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

To ask questions of others who use OSIsoft software, join the OSIsoft user community, PI Square. Members of the community can request advice and share ideas about the PI System. The PI Developers Club space within PI Square offers resources to help you with the programming and integration of OSIsoft products.

Appendix F. Revision History

Date	Author	Comments
13-Jun-2007	Jhartman	Initial Release
18-Jul-2007	Jloe	Version 1.0.1.3 Revision H: updated headers, table formatting, IORates information
26-Jul-2007	Mkelly	Version 1.0.1.3 Revision I: Updated the ICU Control and XML section. Fixed headers and footers. Corrected filenames references for csv files included with distribution.
28-Jun-2008	Jhartman	Version 1.0.2.0, Added /TagConfig argument information and updated version.
10-Oct-2008	Mkelly	Version 1.0.2.0 Revision A; Updated screenshots, replace buffering section, fixed headers and links, updated TOC.
14-Jan-2009	Jhartman	Updated version to 1.0.3.0
02-Feb-2009	Mkelly	Version 1.0.3.0, Revision A; Updated copyright date, added Terminology section, updated TOC.
07-Apr-2009	Jhartman	Changed ranges for EndPoint and PMU Id's to 0-65535 to reflect unsigned short value.
11-Oct-2010	Sbondar	Updated with skeleton Version 3.0.27. Updated the list of tested PMU/PDC
17-Dec-2010	Sbondar	Updated with PDC / PMU Data Source Level Failover and some other additional parameters.
29-Dec-2010	Sbranscomb	Version 1.0.4.x Revision A; Updated with skeleton Version 3.0.31.
28-Jan-2011	Mkelly	Version 1.0.4.x Revision B; Update ICU Control screenshots for new parameters.
10-Feb-2011	Sbondar	Version 1.0.4.x Revision C; Update tables of Attributes and Elements with the new parameters.
15-Feb-2011	Sbondar	Version 1.0.4.x Revision D; Change phrase from 'server-level failover' to 'Data Source Level Failover' to be consistent with other documents. The reason for that is PMU / PDC are considered to be more like a data source rather than a server.
21-Jul-2011	Twilliams	Version 1.0.4.x – 1.0.5.x; Updated the version number for a rebuild with new UniInt 4.5.2.0.
21-Feb-2013	Mkelly	Version 1.0.6.x Updated ICU Control screenshots for new MulticastSourceIPAddress XML Element.
25-Feb-2013	Sbranscomb	Version 1.0.6.x; Revision A; Updated with skeleton Version 3.0.36.
28-Feb-2013	Sbondar	Version 1.0.6.x Revision B; Updated description for new MulticastSourceIPAddress XML Element

Date	Author	Comments
01-Apr-2013	Sbondar	Version 1.0.6.x Revision C; Updated /TagConfig and /configfile parameters in command-line. Parameters table
01-Apr-2013	Sbondar	Version 1.0.6.x Revision D; Updated InstrumentTag attribute description.
01-July-2013	Sbondar	Version 1.0.6.x Revision E; Updated Trademarks page.
13-Apr-2015	WZhang	Version 1.0.6.x Revision F; Update terminology
01-Jun-2015	JMuraski	Version 1.1.0.x Updated with skeleton version 3.0.42
11-Aug-2015	TWilliams	Version 1.1.0.x. Added text to more clearly define how to configure a service using Local System

PI System Explorer

User Guide

For PI Asset Framework 2.10 included with PI Server 2018

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Fax: (01) 510-357-8136
Web: http://www.osisoft.com

PI System Explorer User Guide

© 2009-2018 by OSIsoft, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of OSIsoft, LLC.

OSIsoft, the OSIsoft logo and logotype, Managed PI, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, RLINK, and RtReports are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Version: 2.10

Published: 08 May 2018

Contents

Introduction to PI Asset Framework 1
How assets are represented in PI AF 1
PI System Explorer 2
PI System Explorer user interface components 3
Feature suggestions 4
PI System Explorer browser and navigator 4
Customize the navigator 5
PI System Explorer customization options 5
Column display configuration 7
Select multiple objects in the viewer 9
Object duplication 9
Icons that indicate object states 10
Check-in of database changes 11
Keyboard shortcuts 12
Change the language 13
Valid characters in PI AF object names 14
PI time 14

Server and database connections 19
PI AF server connections 19
Connect to a PI AF server 20
Add a PI AF server to the connection list 21
PI AF server properties 22
View general properties of the connected PI AF server 22
Manage identities 23
Manage mappings 24
View PI AF server object counts 26
PI Data Archive connections 26
Connect to PI Data Archive 26
Add PI Data Archive servers to connected server lists 28
View buffering status for connected PI Data Archive servers 29
PI AF database connections 30
Create PI AF databases 30
View or edit properties of PI AF databases 31
Connect to a database on a different PI AF server 32
Set the default database 33
Rename databases 33
Search for PI AF databases 33
Refresh the list of databases 34
Database deletion 34
PI AF and PI Data Archive collective connections 34
Manage connection preferences for PI System Explorer 34
Switch collective members 37
Switch to a specific collective member 37

Database import and export 39
Export a database to XML 39
XML export options 40

All Referenced Objects.... 40
File format for export and restore.... 43
Restore databases from exported XML files.... 44
Export of library objects to another database.... 45
PI AF saved libraries.... 45
Save databases as libraries.... 46
Load database libraries.... 46
Review properties of loaded libraries.... 46
AFExport utility.... 47
Guidelines for exporting collections.... 47
AFExport utility parameters.... 47
Sample export paths.... 49
AFImport utility.... 50
AFImport utility parameters.... 50

Retrieval of asset information.... 53
Asset browsing.... 53
Element browsing.... 54
Configure browser page size.... 54
Open additional browser windows.... 55
Asset and asset data searches.... 55
Quick search.... 56
Search result paging.... 56
Maximum query sizes.... 57
Searches on a specified date.... 57
Search for PI points.... 57
Manage search results.... 63
Search for elements.... 64
Special characters in name searches.... 69
Search for attributes on elements.... 69
Search for event frames.... 71
Search for attributes on event frames.... 74
Search for transfers.... 76
Search for attributes on transfers.... 77
Trends in PI System Explorer.... 79
Create trends.... 79
Version management for equipment and processes.... 80
Create versions for elements or models.... 80
Create table versions.... 81
Compare two versions.... 81
Display of different object versions and obsolete objects.... 82
Query Date and PI System Explorer time.... 83
Show specific dates and times.... 83
Time context for attribute values.... 84
View time series data.... 85
Boundary type.... 86
Filter expression.... 86
Statistics.... 87
Weighting.... 87
Event horizon mode.... 88
Restrictions on viewing time series data.... 89

Organization of asset models in PI AF.... 93

The structure of PI AF asset models....93
The design of PI AF asset hierarchies....94
Element references in the asset hierarchy....95
Reference types....96
Create single element references....98
Create multiple element references....98
Locate other references to the same element....99
Change reference types....99
Creation of custom reference types....100
Categorization of objects....100
Create new categories....101
Assign objects to categories....102

Representation of assets in PI AF....103
Asset representation with templates....103
Template strategy....104
Element templates....104
Default attribute....105
Base and derived templates....105
Extensions....106
Naming patterns....107
Create element templates....109
Find template references....111
Define templates for other objects....113
Child templates....113
Create child template references....114
Asset representation with elements....114
Create elements based on a template....115
Create elements not based on a template....116
Find where an element is used....116
Deletion of elements....117
Attribute-value reset to original properties....117
Convert elements to element templates....118
Change element templates....118
Storage of application-specific information....119
Create extended properties....119

Association of data with PI AF assets....121
Attribute templates....121
Attribute properties....122
Attribute traits....124
Control the display of attribute and attribute template values....134
Create attribute templates....135
PI AF data types....137
Define constant values for attribute templates....138
Add attribute extensions to template-based elements....140
Enumeration sets....142
Manage enumeration sets....144
Renumber enumeration values....146

Configuration of data references....149
Attribute configuration strings in PI System Explorer....149
Configuration strings for PI point data references....150

Path syntax for references to other attributes... 153
Substitution parameters in data references... 156
Symbols used in substitution parameters... 157
Name substitutions... 157
References to attribute values... 158
List of PI AF substitution parameters... 158
Format strings for time substitution parameters... 162
Find default PI Data Archive servers... 163
PI point data references... 164
Direct PI point references... 165
Indirect PI point references... 165
Create direct PI point data references from Tag Search results... 166
Configure direct or indirect PI point data references... 167
Unit of measure considerations... 168
Configuration of retrieval methods for attribute values... 168
PI point value updates from a data reference... 176
Attribute-value updates from PI point data references... 177
Configure automatic creation of PI points... 178
Edit PI point properties... 180
Preview substitution parameters in PI point data references... 180
PI point array data references... 181
Create PI point array data references... 181
Formula data references... 182
Formula data reference operators... 182
Formula data reference functions... 183
Units of measure in formula data references... 185
Configure formula data references... 185
String Builder data references... 188
Attribute references in String Builder... 188
Function implementation in String Builder... 189
Create String Builder data references... 192
Table lookup data references... 195
Create PI AF tables... 196
Create table column definitions... 198
Populate tables manually... 199
Data references from outside the PI System... 200
Configure table lookup data references... 211
URI Builder data references... 219
Create URI Builder data references... 220

Units of measure in PI AF... 225
Case sensitivity of UOM abbreviations... 226
Configuration of case-sensitive UOM abbreviations... 227
Base classes and derived classes... 228
UOM base classes... 228
Common UOM derived classes... 228
Create UOM classes... 229
Create units of measure... 230
UOM conversion calculation with a formula... 232
Calculate conversion values for a UOM... 233
UOM groups... 233
Manage UOM groups... 234
Origin of UOMs... 235

Security configuration in PI AF.....237
PI AF identities and mappings.....237
Built-in PI AF identities.....239
Security hierarchy.....240
Security inheritance of PI AF objects.....241
Permission inheritance of element objects.....242
PI AF access rights.....246
Deny permission option.....248
Security Configuration window.....249
PI AF server security.....250
Configure security for a PI AF server.....251
PI AF database security.....252
Configure security for new PI AF databases.....252
Configure security for a single PI AF database.....253
PI AF collection security.....254
Configure security for collections.....254
PI AF object security.....256
Configure security for objects.....256
Configure security for analyses and analysis templates.....257
Configure security for the UOM database.....258
Differences from PI Data Archive security model.....258

Event frames in PI AF.....261
Structure of event frames.....262
Advantages of event frames.....262
Examples of event frames.....263
Event Frames or Batch?.....264
Ways to create event frames.....264
Visualization of event frames.....265
Event frame templates.....266
Primary referenced element.....267
Data references to attributes in the primary referenced element.....267
Data references to attributes in other elements.....267
Create event frame templates.....269
Change event frame templates.....272
Create event frames.....273
Create attributes on event frames.....275
Value capture for event frames and transfers.....276
Capture values.....276
Lock event frames or transfers.....277
Acknowledgement of event frames.....278
Acknowledge event frames.....279
Transfers.....279
Create transfer templates.....280
Create transfers.....282
Create transfer ports.....283

Annotations in PI AF.....285
Element annotations.....285
Add annotations to elements.....286
Event frame annotations.....287
Add annotations to event frames.....287

Transfer annotations ... 288
Add annotations to transfers ... 288

Asset analytics ... 291
Introduction to analyses ... 291
Configure and manage analyses on an element ... 292
Analysis templates ... 294
Updates of analyses and analysis templates ... 295
Excluded attributes in analyses ... 295
Analysis scheduling ... 296
Expressions for analyses ... 298
Future data and analyses ... 301
Expression analyses ... 302
Create an expression analysis template ... 302
Create an expression analysis ... 305
Rollup analyses ... 307
Rollup analysis functions ... 308
Create a rollup analysis template ... 308
Create a rollup analysis ... 312
Event frame generation analyses ... 315
Implicit modes of event frame generation in asset analytics ... 316
Create an event frame generation analysis template ... 317
Create an event frame generation analysis ... 320
Start and end trigger conditions ... 322
Child event frame generation with multiple start triggers ... 323
SQC analyses ... 324
Create an SQC analysis template ... 325
Create an SQC analysis ... 327
Sample analyses ... 329
Build an expression analysis to study efficiency deviation ... 330
Create a template for power rollup analyses ... 332
Create event frames automatically to track inefficiency ... 334
Backfilling and recalculation of analyses ... 336
Backfill or recalculate data for an analysis ... 337
Backfill or recalculate data for multiple analyses ... 338
Reconciliation of event frames during backfilling ... 339
Configure automatic recalculation of analyses and analyses templates ... 340
Enable or disable automatic recalculation for multiple analyses ... 341
Open recalculation log folder ... 341
Management of analyses in a PI AF database ... 341
Expression functions reference ... 342
Abs ... 349
Acos ... 350
AND ... 350
ArrayLength ... 351
Ascii ... 351
Asin ... 352
Atn ... 353
Atn2 ... 353
Avg ... 354
BadVal ... 355
Bod ... 356
Bom ... 356

Bonm..... 357
Ceiling..... 358
Char..... 359
Compare..... 359
Concat..... 360
Contains..... 361
Convert..... 362
Cos..... 362
Cosh..... 363
Cot..... 364
Coth..... 364
Cov..... 365
Csc..... 366
Csch..... 367
Curve..... 367
Day..... 368
DaySec..... 369
DeltaValue..... 369
DigState..... 370
DigText..... 371
E..... 371
ELSE..... 372
EventCount..... 372
EventFrame..... 373
Exp..... 374
FilterData..... 375
FindEq..... 375
FindGE..... 376
FindGT..... 377
FindLE..... 378
FindLT..... 379
FindNE..... 380
FirstValue..... 381
Float..... 381
Floor..... 382
Format..... 383
Frac..... 383
HasChanged..... 384
HasValueChanged..... 385
Hour..... 385
IF..... 386
IN..... 387
InStr..... 387
Int..... 388
InterpolatedValues..... 389
IsSet..... 390
LastValue..... 391
LCase..... 392
Left..... 392
Len..... 393
LinRegr..... 393
Log..... 395
Log10..... 395

Logbase....396
LTrim....397
MapData....397
Max....398
Median....399
Mid....400
Min....401
Minute....402
Mod....402
Month....403
NextEvent....403
NextVal....404
Noon....405
NoOutput....405
NormalIrnd....406
NOT....406
NumOfChanges....407
OR....407
ParseTime....408
PctGood....409
Pi....409
Poisson....410
Poly....410
Pow....411
PrevEvent....412
PrevVal....412
PStDev....413
Rand....414
Range....415
Rate....416
RecordedValues....416
RecordedValuesByCount....417
Remainder....419
Right....420
Round....420
Roundfrac....422
RTrim....423
Sec....423
Sech....424
Second....425
SecSinceChange....425
Sgn....426
Sin....426
Sinh....427
Sqr....427
StateNo....428
SStDev....429
StDev....430
String....431
TagAvg....432
TagBad....433
TagDesc....433
TagEU....434

TagExDesc ... 435
TagMax ... 435
TagMean ... 436
TagMin ... 437
TagName ... 438
TagNum ... 439
TagSource ... 439
TagSpan ... 440
TagTot ... 440
TagType ... 442
TagTypVal ... 443
TagVal ... 443
TagZero ... 444
Tan ... 445
Tanh ... 445
Text ... 446
THEN ... 446
TimeEq ... 447
TimeGE ... 448
TimeGT ... 449
TimeLE ... 450
TimeLT ... 451
TimeNE ... 452
TimeStamp ... 452
Total ... 453
Trim ... 454
Trunc ... 455
UCase ... 456
Weekday ... 456
Year ... 457
Yearday ... 457
Steam functions for analysis expressions ... 458
Steam functions in asset and tag-based analytics ... 459
Ranges for steam function inputs ... 460
Unit of measure conversion for steam functions ... 462
Steam property reference states ... 463
Steam functions reference ... 463
Asset analytics native integration of external systems ... 477
Integration of MATLAB Production Server into asset analytics ... 478
Configure an external system endpoint ... 479
Supported functions and data types in MATLAB Production Server ... 479
Error handling ... 480
PI Analysis Service management ... 481
View analysis service statistics ... 481
View or modify analysis service ... 482
Analysis service configuration ... 482
Buffering PI point outputs for PI Analysis Service ... 486

Notifications ... 487
Introduction to notifications ... 488
Notifications and event frames ... 488
Notifications history ... 489
Requirements for notifications ... 490

Client support for notifications 490
Notifications security..... 490
Security for contacts..... 491
Security for notification rules and templates..... 492
Configure authentication for SMTP server connection..... 493
Configure authentication for a web service connection..... 494
Create a notification rule..... 495
Create a notification rule template..... 495
Trigger criteria in notification rules..... 496
Non-Repetition Interval in notification rules..... 498
Resend Interval in notification rules..... 498
Delivery formats in notification rules..... 498
Add a domain to the whitelist..... 500
Add a screenshot in notification email delivery format..... 502
Subscriptions in notification rules..... 503
Customization of subscription content and delivery..... 504
Configuration of notifications delivery endpoints..... 504
Configure an email delivery endpoint..... 505
Configure a dynamic email delivery endpoint..... 506
Configure a REST web service delivery endpoint..... 506
Configure a SOAP web service delivery endpoint..... 508
Contacts..... 509
Configure Active Directory access for contacts..... 511
Manage contacts..... 512
Options for the notifications email delivery channel..... 516
Options for the notifications web service delivery channel..... 516
Configuration of notification rules for analyses or event frames..... 517
Configure notification rules from analyses..... 518
Configure notification rules for user-defined event frames..... 521
Configure escalations for notification rules..... 524
Management of notification rules..... 524
Changes from PI Notifications 2012..... 524
Migration from PI Notifications 2012..... 525
Additional resources..... 528

Management of analyses and notification rules..... 529
Search for analyses or notification rules..... 529
Management of analyses..... 530
Management of notification rules..... 532

Process models in PI AF..... 533
The scope of a PI AF model..... 533
Guidelines for PI AF models..... 534
Submodels..... 534
Element types in models..... 535
Create PI AF models..... 535
Edit PI AF models..... 536
Ports and connections..... 536
Create ports..... 536
Create connections..... 537
Layers..... 537
Create layers..... 538

PI AF utilities and plug-ins **539**
Launch PSE with command line options 539
AFExplorer parameters 539
PI AF Diagnostics utility 540
Grant permissions for PI AF Diagnostics utility 541
Run the AFDiag utility 542
AFDiag utility parameters 543
Audit Trail implementation 548
AF Update Plug-in Configurations utility 549
Set PI AF server utility 551
SetPISystem utility parameters 551
Capture AF SDK event trace output 552
AFGetTrace utility parameters 552
Track PI AF changes with Audit Trail 553
Review changes with the Audit Trail utility 554
View installed plug-ins 556
Command-line plug-in registration 556
RegPlugIn parameters 558
Create an XML registration file 559
Create an SQL registration script 560
Register plug-ins with generated SQL scripts 561
Plug-in provider management 561
Locate the names of trusted providers 562

Technical support and other resources **563**

Introduction to PI Asset Framework

With PI Asset Framework you can represent assets and processes as PI AF objects and structure them to provide value to your business.

PI System Explorer and PI Builder are the primary tools that you use to create and manage PI AF objects. Both tools include support for the following features:

- **Event frames**

  With event frames, you can capture important process events and collect relevant data around them to help analyze why they occurred. See Event frames in PI AF for more information.

- **Asset analytics**

  Integrated into PI System Explorer, you can use asset analytics to create calculations and set up conditional statements involving asset values. See Asset analytics for more information.

- **Notifications**

  Integrated into PI System Explorer in PI AF 2016 R2, you can use notifications to create rules by which users can be alerted in real time to anomalous conditions in the system. See Introduction to notifications for more information.

Topics in this section

- How assets are represented in PI AF
- PI System Explorer

How assets are represented in PI AF

PI Asset Framework (PI AF) enables you to build a representation of your equipment and processes that can give you tremendous insight into your data. In PI AF, the equipment and processes that you want to monitor are called assets. The PI AF representation of all your assets and processes together is called an asset model. The asset model organizes all your equipment into a structure that makes it easy to find information.

In the PI AF asset model, each piece of equipment is represented by an element. The associated data for an element is stored as attributes on the element. Attributes can hold simple values, representing fixed information, such as the diameter of a tank. An attribute can alternatively reference a PI point, a formula, a value from a relational database, a file, a photograph, and more. All relevant data about an asset is tied to the element representing that asset.

For example, suppose you have a pump with three associated pieces of data: the pressure (read from a PI point), the inlet temperature, and the outlet temperature. To model this in PI AF, you can create a PI AF element to represent the pump and then create three attributes to represent the associated data.

The following illustration shows how the data might look in PI AF. Although all the values are PI point values, the user never needs to know the names of the PI points. All the data is available directly on the element.

Name	Value
Head Pressure	92.1994592
Temperature Inlet	15 deg C
Temperature Outlet	16 deg C

Video

For information on assets and attributes, watch this video:

https://www.youtube.com/watch?v=18RgS8Y3JtQ&rel=0

PI System Explorer

PI System Explorer provides a graphical user interface for creating, editing, and managing PI AF objects. Use PI System Explorer to create and manage your asset framework, including PI AF databases, elements, and templates.

PI System Explorer versions

On computers with a 64-bit operating system, both a 32-bit and a 64-bit version of PI System Explorer are available after a PI AF client installation. Beginning with PI AF 2017, the 32-bit version is labeled **PI System Explorer Legacy 32-bit**, whereas the 64-bit version is labeled simply **PI System Explorer**.

The legacy 32-bit version is available should you need to continue to use the legacy version of Notifications (prior to the 2016 R2 version) and its contact editor. This version of the contact editor is required to work with legacy delivery channels. For more information on notifications, see Notifications.

Support for FIPS

Beginning with PI AF 2017 R2, PI AF supports Windows environments that enforce the use of U.S. Federal Information Processing Standards (FIPS) cryptographic algorithms. No additional configuration is required once the local security policy **System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing** has been enabled.

To be in compliance with FIPS, users who need to use passwords to link to external tables can only use PI AF 2017 R2 or later clients and servers. Furthermore, users cannot export data from a PI AF database that contains table passwords created with a PI AF 2017 R2 or later client and import those passwords to PI AF clients on older releases. For more information on linking to external table data references, see Data references from outside the PI System.

Topics in this section

- PI System Explorer user interface components
- Feature suggestions
- PI System Explorer browser and navigator
- Customize the navigator
- PI System Explorer customization options

- Column display configuration
- Select multiple objects in the viewer
- Object duplication
- Icons that indicate object states
- Check-in of database changes
- Keyboard shortcuts
- Change the language
- Valid characters in PI AF object names
- PI time

PI System Explorer user interface components

The following illustration shows the major components of the PI System Explorer.

1. Menu bar
2. Toolbar
3. Browser
4. Navigator
5. Status bar
6. Palette
7. Viewer

Videos

For information on PI System Explorer browser and navigator, watch this video:

https://www.youtube.com/watch?v=Dtf7xC2N7ak&rel=0

For information on PI System Explorer palette and viewer, watch this video:

https://www.youtube.com/watch?v=IXXR88qBv2I&rel=0

Feature suggestions

Beginning with PI AF 2017, you can make suggestions on features you would like to see added to the PI System. You can also review suggestions that other users have already submitted and vote for those you approve of. You can select from the following categories for your suggestion:

- Analytics and Calculations
- Asset Framework
- Data Archive
- Event Frames
- General
- Help/Documentation/Videos
- Installation
- Notifications
- PI Builder
- Security
- System Management

To provide feedback, you select **Help** > **Give Us Feedback** and click **Sign in**. You can type your feedback into the **Enter your idea** field, select an appropriate category, and provide additional details as needed.

PI System Customer Experience Improvement Program

The **Customer Experience Feedback Option** on the **Help** menu enables you to participate in the telemetry program that OSIsoft uses to collect anonymous usage data to improve the PI System and to prioritize new features. The collected data does not include business data or logic, but can include information such as IP addresses, host names, and names of PI System objects.

To participate in the telemetry program, you select **Yes, I would like to participate (Recommended)** and click **OK**.

PI System Explorer browser and navigator

The content of the PI System Explorer browser depends on which button is selected in the navigator. The default available objects in the navigator are elements, event frames, library objects, units of measure, and contacts. The management object is also displayed if the Management plug-in (which encompasses analyses and notification rules) is installed.

> **Note:**
> If legacy notifications are installed on a PI AF client computer, MyPI and Notifications may also be displayed in the navigator.

You can move very quickly between browser objects in PI System Explorer with the following keyboard shortcuts. Notice also that different colors are displayed at the top of the browser and viewer to help distinguish between selected browser objects:

Shortcut	Browser
CTRL+1	Elements
CTRL+2	Event Frames
CTRL+3	Library
CTRL+4	Unit of Measure
CTRL+7	Contacts
CTRL+0	Management

Customize the navigator

To increase the space available on your screen for objects in the browser, change the buttons in the navigator to icons.

Procedure

1. Right-click in the navigator and click **Navigation Pane Options**.
2. In the Navigation Pane Options window, clear the check box beside each component on the **Display Full Button** list.
3. Click **OK**. The rows of button labels are replaced by a single row of icons.

4. Move your cursor over an icon to display its label.

PI System Explorer customization options

You can customize many aspects of PI System Explorer to suit your needs with **Tools** > **Options**.

General tab

You use the **General** tab to control display options for several features. You can control:

- Keystroke to open Check-In and Undo-Checkout windows. For more information, see Check-in of database changes.
- Title bar appearance.
- Page size for browser objects. For more information, see Configure browser page size.
- Number of queries returned in object searches. For more information, see Search result paging.

- Unit of measure appearance for attributes. For more information, see Show attribute values in source unit of measure.
- Display of attribute values to the units of measure that are mapped to a selected UOM group. When no UOM group is selected, attribute values are based on the default UOM defined in attribute templates or their source unit of measure (if **Use Source Unit-Of-Measure for attribute display** is selected). For more information, see UOM groups.
- Display of excluded attributes. For more information, see Excluded attribute property.
- Number of digits displayed for attribute values. For more information, see Control the display of attribute and attribute template values.

Time Context tab

You use the **Time Context** tab to define the time or time range that PI System Explorer uses to display attribute values. For more information, see Set time context for displayed attribute values.

Server Options tab

You use the **Server Options** tab to define how PI System Explorer should connect to a PI AF collective or PI Data Archive collective. For more information, see Manage connection preferences for PI System Explorer.

Topics in this section

- Show attribute values in source unit of measure
- Change the unit of measure for displayed attribute values

Show attribute values in source unit of measure

For attributes that are defined by data references, you can have their values displayed in the units of measure that are defined by the data reference, rather than in the default units of measure that are established in attribute templates.

Procedure

1. Select **Tools** > **Options**.
2. On the **General** tab, select **Use Source Unit-Of-Measure for attribute display**.
3. Click **Apply**.
4. Click **OK** to close the window.

Change the unit of measure for displayed attribute values

During your current session in PI System Explorer, you can change the unit of measure (UOM) that is displayed for an attribute value from its default or mapped value.

Procedure

1. In the **Attributes** tab of the **Elements** or **Event Frames** viewer, right-click the value for an attribute and click **Change Display UOM**.
2. In the Select Display UOM window, select a new UOM for the value from the **Display UOMs** list.

   Note:
   The list contains all UOM values in a class. **<Default>** is displayed beside the default UOM. If you have selected a UOM group in **Tools** > **Options** and the selected UOM group has a mapping for the attribute's default UOM, **<Mapped>** is displayed beside the mapped UOM.

3. Click **OK**.

Column display configuration

Property columns

Whenever data columns are displayed in the viewer or a search results window, you can customize which property columns you wish to view. At the far right of the window, you can click and select additional column selections or clear default column selections.

Attribute columns

In viewer or search results windows for elements and event frames, you can also select attribute columns and display them as columns alongside the default property columns. An attribute column is identified by the **|** character.

Select attributes to display as viewer columns

For collections of **Elements**, **Element Searches**, **Event Frames Searches**, and **Transfer Searches**, you can select attributes to display as columns in the viewer, in addition to default properties.

Procedure

1. In the navigator pane, choose one of the following actions:
   - To select attributes for **Elements** and **Element Searches** collections, click **Elements**.
   - To select attributes for **Event Frames Searches** and **Transfer Searches** collections, click **Event Frames**.
2. In the browser, click the collection for which you want to select attributes.
3. At the far right of the viewer, click .
4. Click **Select Attributes**.
5. In the Select Attributes window, choose from the following options to select the attributes that you want displayed in the viewer:

To ...	Do this ...
Add attributes from a template	a. Select the **Add Attributes from Template** option. b. Select a template from the list.
Add attributes from an element or event frame **Note:** You cannot add attributes from a **Transfer Searches** collection	a. Select the **Add Attributes from Element/ Event Frame** option. b. Choose from the following actions: ▪ To search for an element or event frame, click and enter criteria to locate attributes in the Element Search or Event Frame Search window. ▪ To select an element or event frame from the browser tree, click and select from the Element Browser or EventFrame Browser window. c. Click **OK**.
Add other attributes	a. In the **Others** field, enter attribute names separated by a semicolon, or click and enter criteria to locate attributes in the Attribute Search window. b. Click **Add**.

Selected attributes are displayed in alphabetical order in the left-hand **Attributes** or **Attribute Templates** list. For added convenience, you can group them by category or template by selecting one of the **Group by** check boxes. You can also filter them if necessary with further search criteria.

6. From the **Attributes** or **Attribute Templates** list, select the attributes you want to display in the viewer.
   a. Use standard Windows selection keystrokes (such as SHIFT+<click> and CTRL+<click>) to select contiguous and non-contiguous attributes in the list, as described in Select multiple objects in the viewer.
   b. Click to move selected items or to move all items to the right-hand **Attributes** list.
7. Optional. Adjust the order that attributes will be displayed in viewer columns:

To ...	Do this ...
Move an attribute up the list	Select an attribute below the top row and click .
Move an attribute down the list	Select an attribute above the bottom row and click
Remove an attribute from the list	Select an attribute and click .
Remove all selected attributes	Click .

8. Optional. You can add additional attributes as required, for example from another template. Repeat steps 5 to 7.
9. Click **OK** to complete the attribute selection.

The selected attributes are displayed as columns in the viewer.

10. Optional. You can control which selected attribute columns are displayed in the viewer:

To ...	Do this ...
Remove an attribute column	a. Click ⚙. b. Click **Attribute Columns**. c. Clear the attribute you want to remove.
Remove all attribute columns	a. Click ⚙. b. Clear **Show Attribute Columns**.

Select multiple objects in the viewer

In the viewer, use standard Windows mouse and keyboard combinations to select more than one object to complete operations such as copying, deleting, exporting, checking items in and out, as well as changing templates.

Procedure

- To select a group of contiguous objects in the viewer, click one object, move the cursor to the end of the group, press SHIFT and click the last object in the group.
- To select non-contiguous objects, press the CTRL key as you click each object in the viewer.
- To select all objects in the viewer, press CTRL+A. To deselect individual objects in the selection, press CTRL and click each object.

Results

After selecting the objects, right-click and click the operation to perform on the context menu.

Object duplication

You can copy individual objects or multiple rows of objects and paste the information for those objects into a spreadsheet in a tab-separated format.

The **Copy** option is available on context menus throughout PI AF. The **Copy Path** and **Copy Cell** options are available where appropriate.

> **Note:**
> When you drag and drop, the clipboard always contains the path information, which renders the data compatible with other client applications, such as PI ProcessBook.

Copy

In any PI AF browser, use **Copy** on a context menu to place the full path of the selected object on the clipboard:

```
\\Kaboom\Chocolate Milk Tutorial\ChocolateMilkModel\MixFlow
```

In any viewer grid or list, use **Copy** to place the content of each displayed column in the selected row on the clipboard. If you select multiple PI AF rows, **Copy** places multiple data

rows separated by a new line. In attributes lists, the copied data includes strings that correspond to the Configuration (), Quality (), Template (), and Hidden () column icons, in addition to other displayed columns:

```
1          2  3    4        5  6        7
Is Not Configuration Item Good Tank Is Not Hidden Level 0 ft MilkTank2|Level
```

1	
2	
3	
4	
5	**Name**
6	**Value**
7	**Path**

Copy Path

The **Copy Path** option places the full path for each selected object on the clipboard. If multiple PI AF objects are selected, **Copy Path** places multiple paths separated by a new line:

```
\\ABACUS-CURRENT\NuGreen\NuGreen\Houston\Cracking Process\Equipment\F-321|Model
\\ABACUS-CURRENT\NuGreen\NuGreen\Houston\Cracking Process\Equipment\F-321|Motor
Amps
```

Copy Cell

In any viewer grid or list containing multiple columns, such as attributes, referenced elements, child event frames, and so on, the **Copy Cell** option places the contents of the selected cell on the clipboard.

Icons that indicate object states

PI System Explorer uses the following (mini) icons superimposed over object icons, such as attributes (), elements (), event frames (), and servers (), to indicate the current state of those objects. Such icons can also be combined to indicate multiple states: for example, uses the referenced and default icons to indicate the primary referenced element in an event frame, whereas uses the connected and default icons to indicate the default connected PI AF server.

Icon	Object state
	Is changed, but not yet applied.
	Is checked out by you.
	Is checked out by a different user.
	Is available in multiple versions.
	Is a reference to another object.
	Is the default.

Icon	Object state
	Is disconnected, such as a collective member.
	Is connected, such as a PI AF server.
	Is a search.
	Is a template.
	Is arranged by category.
	Is unsaved, such as a search.
	Is unsupported, such as a server version.

Check-in of database changes

Whenever you make a change to an object, including whenever you create a new object, PI System Explorer checks that object out from the database. You need to save these changes in the database, although you can apply a change to a selected object by clicking ✓ in the toolbar and then check in your changes later.

- Unsaved changes are indicated by a red check mark icon.
- Changes that have been applied, but not saved to the database are indicated by a dark red check mark icon.

As you work, you can choose from these options to save your changes:

- Click **File** > **Check In**.
- On the toolbar, click **Check In**.
- In the browser, right-click an object and click **Check In** to save changes in the database for that object only.

  The first two options allow you to save changes for all modified objects. The Check In window opens and displays objects that have been modified. You can select them all in or select some to check in, and allow others to remain checked out. Click **Session** to select objects modified only in the current session. You can still check in objects modified in other sessions.

Undo Check Out

If you decide not to save your changes, you can use the **File** > **Undo Check Out** option. In the Undo Check Out window, you clear any items in the list you *do* want to check in, and click **Undo Check Out**.

Undo Check Out for all users

Occasionally, you may be unable to modify a PI AF object because the element is checked out to another user. You see the following message types that inform you that an object is locked by a user:

- In notifications, when you attempt to stop, rename, delete, or otherwise change a notification, the error message `Notification 'Notification_name' with Unique ID 'ID' is locked by user Windows_account` is displayed.
- In PI AF, when you attempt to work with an object, the error message `AF Object Name 'Object_name' with UniqueID 'ID' is locked by user Windows_account` is displayed.

To undo the lock, a user who has the Admin permission on the object set to Allow can select **Show All Users** in the Undo Check Out window, select **All** and then click **Undo Check Out**. For more information, see PI AF access rights.

Videos

For information on the check in/check out mechanism of a PI AF object, watch this video:

https://www.youtube.com/watch?v=Pb8ZKxOaTsE&rel=0

For information on how to check in modifications to PI AF objects, watch this video:

https://www.youtube.com/watch?v=Ky1ICZwwBqY&rel=0

For information on how to undo check out of PI AF objects, watch this video:

https://www.youtube.com/watch?v=OPSy_HbvAyY&rel=0

Keyboard shortcuts

PI System Explorer provides keyboard shortcuts for navigation and other tasks that require a mouse or other pointing device.

Shortcut	Action
CTRL+A	Selects all objects in the viewer.
CTRL+C	Copies the selected object to the clipboard.
CTRL+ALT+C	Copies the path of the selected object to the clipboard.
CTRL+V	Pastes the object on the clipboard to the viewer.
CTRL+X	Cuts (deletes) the selected object and copies it to the clipboard.
DELETE	Deletes the selected object.
SHIFT+DELETE	Same as CTRL+X
INSERT	Adds a new object to the viewer or browser.

Shortcut	Action
HOME	Selects the first row in the viewer, for example, the first row in a table of attributes.
END	Selects the last row in the viewer.
CTRL+HOME	Selects the first cell of the current page in the viewer.
CTRL+END	Selects the last cell of the current page in the viewer.
ALT+HOME	Selects the first page of objects in the viewer.
ALT+END	Selects the last page of objects in the viewer.
CTRL+PAGE UP	Selects the previous page of objects in the viewer.
CTRL+PAGE DOWN	Selects the next page of objects in the viewer.
CTRL+ENTER	If the viewer contains multiple pages of objects, displays the Select Page Number window.
ALT+ENTER	In the browser, displays the properties of the selected object.
SPACE or ENTER	Presses the currently selected button.
Left, Right, Up, and Down Arrows	Navigate objects in the viewer or browser.
F2	Edits the selected object on the viewer. For complex objects, displays the edit window for the object.
F4 or ALT+Up Arrow or ALT+Down Arrow	Displays the choices in the selected list box. For layered lists, displays the complete hierarchy of choices.
CTRL+1	Navigates to the Elements browser.
CTRL+2	Navigates to the Event Frames browser.
CTRL+3	Navigates to the Library browser.
CTRL+4	Navigates to the Unit of Measure browser.
CTRL+5	Navigates to the MyPI browser.
CTRL+6	Navigates to the Notifications browser.
CTRL+7	Navigates to the Contacts browser.
CTRL+0	Navigates to the Analyses browser.

Change the language

You can change the language for PI System Explorer on your computer if you have a Language Pack and the desired language resources have been installed. The language setting is per user

locale, so if others want to use PI System Explorer on the same computer under a different login, they can use different language resources if available.

Procedure

1. Run the Language Pack and select the language resources you want to install, if they are not already available.
2. Click **View** > **Language Settings**.
3. In the Language Settings Tool window, select the target language and click **OK**.
4. Quit and restart PI System Explorer.

   PI System Explorer appears in the specified language.

   Note:
   The language setting is account-specific, so users who log in under a different account see the language specified for that account.

Valid characters in PI AF object names

PI AF has the following constraints on object names:

- Leading or trailing spaces are removed from names.
- Maximum name length is 259 characters.
- Blank names are not permitted.

You can use standard keyboard characters, with the following exceptions:

Object type	Invalid characters
Other than contacts	Control characters plus: * ? ; { } [ ] | \ ` ' "
Contacts	Control characters plus: * ? ; [ ] | \ "

PI time

You can use a special syntax, called PI time, to specify inputs for time stamps and time intervals. PI time uses specific abbreviations, which you combine to create time expressions.

Topics in this section

- PI time abbreviations
- PI time expressions
- Time-stamp specification
- Time-interval specification

PI time abbreviations

When specifying PI time, you can use specific abbreviations that represent time units and reference times.

Time-unit abbreviations

Abbreviation	Full version	Plural version	Corresponding time unit
`s`	`second`	`seconds`	Second
`m`	`minute`	`minutes`	Minute
`h`	`hour`	`hours`	Hour
`d`	`day`	`days`	Day
`mo`	`month`	`months`	Month
`y`	`year`	`years`	Year
`w`	`week`	`weeks`	Week

To specify time units, you can specify the abbreviation, the full version, or the plural version of the time unit, such as `s`, `second`, or `seconds`. You must include a valid value with any time unit. If specifying seconds, minutes, or hours, you can specify a fractional value, such as `1.25h`. You cannot specify fractional values for other time units.

Reference-time abbreviations

Abbreviation	Full version	Corresponding reference time
`*`		Current time
`t`	`today`	00:00:00 (midnight) of the current day
`y`	`yesterday`	00:00:00 (midnight) of the previous day
`sun`[1]	`sunday`	00:00:00 (midnight) on the most recent Sunday
jun[2]	`june`	00:00:00 (midnight) on the current day in June of the current year
`dec` *DD*	`december` *DD*	00:00:00 (midnight) on the *DD*th day of December in the current year
YYYY		00:00:00 (midnight) on the current day and month in year *YYYY*
M-D or *M/D*		00:00:00 (midnight) on the *D*th day of month *M* in the current year
DD		00:00:00 (midnight) on the *DD*th day of the current month

[1]: Use the first three letters as an abbreviation for any day of the week: `sun`, `mon`, `tue`, `wed`, `thu`, `fri`, or `sat`.

[2]: Use the first three letters as an abbreviation for any month of the year: `jan`, `feb`, `mar`, `apr`, `may`, `jun`, `jul`, `aug`, `sep`, `oct`, `nov`, or `dec`.

PI time expressions

PI time expressions can include fixed times, reference-time abbreviations, and time offsets. A time offset indicates the offset direction (either + or -) and the offset amount (a time-unit abbreviation with a value).

For example, PI time expressions can have the following structure:

Structure	Example
Fixed time only	`24-aug-2012 09:50:00`
Reference-time abbreviation only	`t`
Time offset only	`+3h`
Reference-time abbreviation with a time offset	`t+3h`

Include at most one time offset in an expression; including multiple time offsets can lead to unpredictable results.

Time-stamp specification

To specify inputs for time stamps, you can enter time expressions that contain:

- Fixed times

  A fixed time always represents the same time, regardless of the current time.

Input	Meaning
`23-aug-12 15:00:00`	3:00 p.m. on August 23, 2012
`25-sep-12`	00:00:00 (midnight) on September 25, 2012

- Reference-time abbreviations

  A reference-time abbreviation represents a time relative to the current time.

Input	Meaning
`*`	Current time (now)
`3-1` or `3/1`	00:00:00 (midnight) on March 1 of the current year
`2011`	00:00:00 (midnight) on the current month and day in the year 2011
`25`	00:00:00 (midnight) on the 25th of the current month
`t`	00:00:00 (midnight) on the current date (today)
`y`	00:00:00 (midnight) on the previous date (yesterday)
`tue`	00:00:00 (midnight) on the most recent Tuesday

- Reference-time abbreviations with a time offset

  When included with a reference-time abbreviation, a time offset adds or subtracts from the specified time.

Input	Meaning
`*-1h`	One hour ago
`t+8h`	08:00:00 (8:00 a.m.) today
`y-8h`	16:00:00 (4:00 p.m.) the day before yesterday
`mon+14.5h`	14:30:00 (2:30 p.m.) last Monday
`sat-1m`	23:59:00 (11:59 p.m.) last Friday

- Time offsets

Entered alone, time offsets specify a time relative to an implied reference time. The implied reference time might be the current clock time or another time, depending on where you enter the expression.

Input	Meaning
`-1d`	One day before the current time
`+6h`	Six hours after the current time

Time-interval specification

Time-interval inputs define intervals for collecting or calculating values during a time period. For example, you might specify a 60-minute interval to compute an hourly average over a 12-hour period. To specify time-interval inputs, enter a valid value and time unit:

- Positive values define intervals that begin at the earlier time in the period and that finish at or before the later time in the period.

Start time	`2:00:00`
End time	`3:15:00`
Time interval	`30m`
Returned intervals	2:00:00 to 2:30:00 2:30:00 to 3:00:00

- Negative values define intervals that finish at the later time in the period and that begin at or after the earlier time in the period.

Start time	`2:00:00`
End time	`3:15:00`
Time interval	`-30m`
Returned intervals	2:15:00 to 2:45:00 2:45:00 to 3:15:00

Server and database connections

After a PI AF Client installation, an administrator needs to configure a PI AF server for general use. Configuration includes the following tasks:

- Connect to a PI AF server and set up a default.
- Set up PI AF identities.
- Map Windows user accounts to one or more PI AF identities.
- Select a default PI Data Archive, if one was not selected during PI AF Client installation.
- Create (or import) PI AF databases.
- Specify PI System Explorer connection preferences, if PI AF collectives are defined.
- Configure access permissions for identities to the PI AF Client server and databases. For more information, see Security configuration in PI AF.

Video

For information on how to connect to and search a PI System, watch this video:

https://www.youtube.com/watch?v=_n3yLpjMhew&rel=0

Topics in this section

- PI AF server connections
- PI Data Archive connections
- PI AF database connections
- PI AF and PI Data Archive collective connections

PI AF server connections

An administrator needs to select the PI AF servers to which to connect and establish the default. Additionally, an administrator needs to set up PI AF identities and mappings to PI AF servers.

Topics in this section

- Connect to a PI AF server
- Add a PI AF server to the connection list
- PI AF server properties
- View general properties of the connected PI AF server
- Manage identities
- Manage mappings
- View PI AF server object counts

Connect to a PI AF server

Use the Servers window to review PI AF server connections and to connect to different servers as needed.

Procedure

1. From the PI System Explorer menu bar, choose **File** > **Connections**.

   In the Servers window, you see a list of known servers, identified by the following icons.

	A disconnected PI AF server in the Known Server Table.
	A connected PI AF server.
	The default connected PI AF server.
	A disconnected PI AF collective.
	A connected PI AF collective.
	A PI Data Archive in the Known Server Table.
	The default connected PI Data Archive.
	A disconnected PI Data Archive collective.
	A connected PI Data Archive collective.

> **Note:**
> The icon and a warning indicates that the PI AF client is connected to an unsupported version of PI Data Archive, such as version 3.4.375 or 3.4.370. The PI AF client cannot connect to a PI Data Archive server that is running software earlier than version 3.4.370.

2. To connect to a PI AF server, choose from the following actions:

To ...	Do this ...
Connect to a different PI AF server	Select a server in the list and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server in the list and click **Connect As**. b. In the Connect to PI AF Server *server* window, enter a Windows user account name and password. c. Click **OK**.
Connect to a server that is not displayed on the Known Servers Table	Click **Add Asset Server**. For more information, see Add a PI AF server to the connection list.
Set a default PI AF server	Right-click a connected PI AF server in the list and click **Set as Default AF Server**. A icon is displayed beside the server that is now the default.

Connect to a collective	a. Right-click a collective in the list. b. Choose one of the following actions: ▪ To connect to the primary server in the collective, click **Connect to Primary**. ▪ To connect to a specific collective member, click **Connect to Collective Member**, as described in Switch to a specific collective member.

3. Click **Close**.

Add a PI AF server to the connection list

If the PI AF server you want to connect to is not currently displayed on the Known Servers Table, you can add it in the PI AF Server Properties window.

Procedure

1. In PI System Explorer, click **File** > **Connections**.
2. Click **Add Asset Server**.
3. In the PI AF Server Properties window, enter the PI AF server properties.
4. The **Name** field does not have to match the host name. After you connect to a PI AF server, you can change the server name by clicking **Rename** and entering a new name.

   Caution:
   Renaming the PI AF server impacts all clients.

5. The **Account** field is typically not needed in normal installations and should be left blank.

   By default, PI AF clients such as PI System Explorer attempt to use a Service Principle Name (SPN) registered by the PI AF server to establish a secure connection to the server. If an SPN cannot be created by the PI AF server, a secure connection can be established via Microsoft NTLM if the server is running under a system account, such as Network Service. In each case, the **Account** field is not needed and should be left blank.

   If the PI AF server is run under a domain account, however, and an SPN is not successfully created by either the PI AF server or an administrator, the domain account of the server must be specified so that the PI AF client can securely identify the server.

6. The default **Timeout** value of 300 seconds is acceptable in most cases. If you experience timeout errors as you work in PI System Explorer, increase the time in the **Timeout** box.
7. Optional. **Aliases** are alternate names for the PI AF server which users can use to look for the server. When configured, PI AF server aliases are stored locally on the client only.
8. Optional. The **Configure Active Directory** link is for setting up the notifications contacts list. This is a PI AF system administrator function.
9. Click **Connect**.

   Note:
   If an error message opens saying that you cannot connect to the PI AF server, you need to enter the domain account of the server in the **Account** field.

10. After a server if connected, you can click **Security** and set up security for the server. For more information, see Security hierarchy.
11. Click **OK**.

PI AF server properties

You manage the configuration of the currently connected PI AF server in the PI AF Server Properties window. The following tabs are available:

- **General**. For more information, see View general properties of the connected PI AF server.
- **Plug-Ins**. For more information, see View installed plug-ins.
- **Libraries**. For more information, see Review properties of loaded libraries.
- **Identities**. For more information, see Manage identities.
- **Mappings**. For more information, see Manage mappings.
- **Counts**. For more information, see View PI AF server object counts.

View general properties of the connected PI AF server

On the **General** tab of the PI AF Server Properties window, review and modify settings for the currently connected PI AF server, such as:

- Server name, host name, and ID
- Server account
- Server version number
- Server port, timeout, and aliases

Procedure

1. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
2. On the **General** tab of the PI AF Server Properties window, review the server information and modify as needed.
3. You can also choose from the following actions.

To ...	Do this ...
Rename the PI AF server	a. Click **Rename**. b. In the **New Name** field of the New PI AF Server Name window, replace the current name with a new name. c. Click **OK**. **Caution:** Renaming the PI AF server impacts all PI AF clients.

Enter an alias	a. Click . b. In the **New Name** field of the New Alias Name window, enter an alternate name that users can use to look for the PI AF server. Aliases are stored locally on the client.
Configure Active Directory access for contacts	See Configure Active Directory access for contacts.
View or modify security permissions of mapped user identities	a. Click **Security**. b. In the Security Configuration window, review the permissions for the listed identities that have been configured for each server component. You can add and remove identities (as well as create new identities and mappings) as needed. For more information, see Configure security for a PI AF server.

4. Click **OK**.

Manage identities

Use the **Identities** tab to review identities that are currently assigned to the PI AF server. You can also add, remove, and edit identities.

Procedure

1. On the PI System Explorer toolbar, click .
2. In the PI AF Server Properties window, click the **Identities** tab.
3. Review the alphabetized list of identities. Click a column heading to change the sort order. Enabled identities are indicated as **True**, whereas disabled identities are grayed out and indicated as **False**. The built-in identities **Administrator** and **World** are read only.
4. You can choose from the following actions:

To ...	Do this ...
Disable an identity	a. Double-click an enabled identity on the list. b. In the Identity Properties window, clear the **Identity is Enabled** checkbox. c. Click **OK**.
Enable an identity	a. Double-click a disabled identity on the list. b. In the Identity Properties window, select the **Identity is Enabled** checkbox. c. Click **OK**.
Create a new identity	a. Right-click and select **New Identity**. b. In the Identity Properties window, enter a name and description for the identity. c. Optional. Click the **Mappings** tab and click **Add**. For more information, see Add a mapping to an identity.

Manage security settings for identities	a. Right-click a column title or below the list and click **Security**. b. In the Security Configuration window, you can review or modify permissions for a listed identity, as well as add or remove identities as needed. For more information, see Configure security for a PI AF server.
Delete an identity	a. Right-click an identity on the list and click **Delete**. b. In the Delete confirmation window, click **Yes**.
Rename an identity	a. Right-click an identity on the list and click **Rename**. b. Enter a new identity name. c. Right-click and click **Check In** to apply the change.

Manage mappings

Use the **Mappings** tab to review mappings that are currently set up on the PI AF server. You can also add, remove, and edit mappings.

Procedure

1. On the PI System Explorer toolbar, click .
2. In the PI AF Server Properties window, click the **Mappings** tab.
3. Review the alphabetized list of mappings. If necessary, scroll rightward to see mapped identities and Windows accounts.
4. You can choose from the following actions:

To ...	Do this ...
Modify the properties of a mapping	a. Double-click a mapping on the list. b. In the Mapping Properties window, you can edit the name and description as needed, as well as select a different identity from the **Identity** drop-down list. c. Click **OK**.
Create a new mapping	See Add a mapping to an identity.
Delete a mapping for an identity	a. Right-click a mapping on the list and click **Delete**. b. In the Delete confirmation window, click **Yes**. **Note:** If your user account is associated with the mapping you are deleting, a warning that you might be locked out is displayed.

Rename a mapping	a. Right-click a mapping on the list and click **Rename**. b. Enter a new mapping name. c. Right-click and click **Check In** to apply the change.
Manage security settings for mappings	a. Right-click a column title or below the list and click **Security**. b. In the Security Configuration window, you can review or modify permissions for a listed identity, as well as add or remove identities as needed. For more information, see Configure security for a PI AF server.

Add a mapping to an identity

When you add a mapping for an identity, you need to specify the Windows account to be mapped and the name of the mapping.

Before you start

The security identifier (SID) for the account is generated automatically and cannot be changed while the mapping is being created.

Procedure

1. In the **Account** field of the Mapping Properties window, enter one of the following object types:
   - User
   - Group
   - Built-in security principal
   - Service account
   - Computer name

   You can also click and enter search criteria in the Select User, Computer, Service Account, or Group window to select the object type, location, and name. The available search locations are the local computer hosting the connected PI AF server and active directory nodes that the user can access.

   Note:
   You can also enter a SID directly in the **Account** field. When the entry is validated, a domain identifier is prepended if the account is a domain account. If the account is a local account on the PI System's host machine, the fully qualified domain name is prepended.

2. To validate your entry in the **Account** field, press Tab or click in another field.

   The **Account SID** field displays the SID of the account, and the **Name** field defaults to displaying the account name.

3. In the **Name** field, modify the default account name as needed.

4. Text you enter in the **Description** field is displayed in the PI AF Server Properties window, so you can use this field to differentiate between mappings for any given identity (since in PI AF a single user can be mapped to multiple identities).
5. If an identity has not been selected already, select the identity you wish to associate with the mapping from the **Identity** drop-down list.
6. Click **OK**.

View PI AF server object counts

You can view object counts for the PI AF server to which you are currently connected, such as the number of databases, elements, element templates, event frames, and so on.

Procedure

1. In PI System Explorer, click **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Counts** tab.

PI Data Archive connections

An administrator needs to select a default PI Data Archive, if one was not selected during PI AF Client installation.

Topics in this section

- Connect to PI Data Archive
- Add PI Data Archive servers to connected server lists
- View buffering status for connected PI Data Archive servers

Connect to PI Data Archive

Use the Servers window to review PI Data Archive known server connections and to connect to different servers as needed.

Procedure

1. Click **File > Connections**.

   In the Servers window, you see a list of known servers, identified by the following icons.

	A disconnected PI AF server in the Known Server Table.
	A connected PI AF server.
	The default connected PI AF server.
	A PI Data Archive in the Known Server Table.
	The default connected PI Data Archive.

> **Note:**
> A warning icon ( ) beside a server indicates that PI Data Archive is running an unsupported version, such as version 3.4.375 or 3.4.370. The PI AF client cannot connect to a PI Data Archive server that is running software earlier than version 3.4.370.

2. To connect to a PI Data Archive, choose from the following actions:

To ...	Do this ...
Connect to a different PI Data Archive	Select a server in the list and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server in the list and click **Connect As**. b. In the Connect to PI Data Archive *server* window, enter a Windows user account name and password. c. Click **OK**.
Connect to a server that is not displayed on the Known Servers Table	Click **Add Data Server**. For more information, see Add PI Data Archive servers to connected server lists.
Set a different server as the default PI Data Archive	Right-click a server in the list and click **Set as Default Data Archive**. A icon is displayed beside the server that is now the default.

3. Click **Close**.

Topics in this section

- PI Data Archive connection problems
- Configure clients to allow explicit login prompts

PI Data Archive connection problems

Windows Authentication only uses identity mappings and does not utilize trusts defined with windows credentials for authentication or any other trusts that may be configured. When connecting to PI Data Archive, error messages might appear if there is no PI mapping configured and one of the following conditions exists:

- You are using a PI username and password to login to PI Data Archive.
- You are using the default user with a blank password to login to PI Data Archive.

These errors occur because PI AF 2012 and later has more stringent security settings than earlier versions of PI AF. Clients that connect to PI Data Archive through PI AF 2012 and later are subject to the following behaviors:

- The default behavior for connections is to not prompt the user for an explicit login unless the **Allow login prompt** setting is configured. This setting controls what happens when PI Data Archive authentication fails. If the **Allow login prompt** setting is enabled, a window opens prompting the user for a username and password. If the setting is off, an error message is displayed. The setting is specific to each client computer. For more information, see Configure clients to allow explicit login prompts.
- Default user connections are no longer supported.

To resolve error conditions, configure PI mappings for users connecting to PI Data Archive. For more information, see the PI System Management Tools topic "Identities, Users, and Groups" in Live Library (https://livelibrary.osisoft.com).

PI Data Archive connection error messages

Error Message	Connect Failure Conditions
Insufficient privilege to access PI Data Archive. No Access - Secure Object Default User connections to PI Data Archive are currently enabled from this client but are not supported from PI AF.	Default User connections are enabled in PI SDK.
Insufficient privilege to access PI Data Archive. No supported authentication methods are enabled. No Access - Secure Object Default User connections to PI Data Archive are currently enabled from this client but are not supported from PI AF.	Default User connections are the only authentication method enabled in PI SDK.
Insufficient privilege to access PI Data Archive. No Access - Secure Object The explicit login prompt is not configured on this computer and therefore not allowed.	Explicit logins setting was not set in registry.

Configure clients to allow explicit login prompts

If all configured protocols fail, use the PI SDK Utility to allow a login prompt to be displayed.

Procedure

1. Open **PISDKUtility** from the Windows Start menu.
2. Click **Connections** > **Options**.
3. In the Connection Options window, select the **Allow login prompt (if all configured protocols fail)** check box.
4. Click **OK**.
5. Click **File** > **Exit PISDKUtility**.

Add PI Data Archive servers to connected server lists

If the PI Data Archive server you want to connect to is not currently displayed on the Known Servers Table, add it in the PI Data Archive Properties window.

Procedure

1. Click **File** > **Connections**.
2. In the Servers window , click **Add Data Server**.
3. In the PI Data Archive Properties window, enter properties as needed.

4. Optional. You can change the default name in the **Name** field:
   a. Click **Rename**.
   b. In the **New Name** field in the New PI Data Archive Name window, enter a name for the new PI Data Archive. The name does not need to match the host name.
   c. Click **OK**.
5. Enter the host name in the **Host** field. You can enter:
   - A fully qualified domain name.
   - A server name.
   - An IP address. An IPv6 address must be enclosed in brackets [ ].
6. Unless your particular application requires a different port, accept the default value in the **Port** field.

   Note:
   You can modify the host name and port only when disconnected from the server.
7. The default values for the **Connection Timeout** and **Data Timeout** fields are acceptable in most cases. If you experience connection timeouts when connecting to PI Data Archive from PI System Explorer, increase the time in the **Connection Timeout** field. If you experience timeout errors while accessing PI Data Archive data, increase the time in the **Data Timeout** field.
8. Optional. Enter an alias name in the **Aliases** field.

   Aliases are alternate names that can be used for PI Data Archive. PI Data Archive aliases are stored locally on the client only.
9. Click **Connect** to connect to PI Data Archive.

   Note:
   The **ID**, **Time Zone** and **Version** fields are not editable. **ID** is the PI Data Archive ID, **Time Zone** is the local time zone of the PI Data Archive and **Version** is the PI Data Archive version.
10. Click **OK**.

View buffering status for connected PI Data Archive servers

If PI Buffer Subsystem 4.3 or later is installed and running, you can view the buffer status of a connected PI Data Archive server.

Procedure

1. Click **File** > **Connections**.

   In the Servers window, each PI Data Archive server for which a connection is configured displays a buffering status in the **Buffer Status** column.

   Note:
   PI Buffer Subsystem versions 3.4.380 and earlier display a status of **Unknown**.
2. To view more details and manage buffering for a specific PI Data Archive, click **Buffering Manager** to open the Buffering Manager tool.

> **Note:**
> You can also open Buffering Manager from the PI System Explorer **Tools** menu.

PI AF database connections

PI AF stores the asset framework objects (elements, templates, and so on) in PI AF databases. You can have multiple databases in PI AF, although you can connect to only one at a time. You typically work with PI AF databases in PI System Explorer or in PI Builder. When you start PI System Explorer, it connects to the default PI AF database. If no databases are defined, PI System Explorer prompts you to create a new database.

> **Note:**
> The **Configuration** database is used by PI AF clients, such as PI Web API, as they interact with PI AF. Do not use this database for your own application data.

Videos

For information on how to create PI AF databases, watch this video:

https://www.youtube.com/watch?v=P7Zopif-j-c&rel=0

For information on how to work with multiple PI AF databases, watch this video:

https://www.youtube.com/watch?v=CN0R7cW3BUc&rel=0

For information on the configuration database, watch this video:

https://www.youtube.com/watch?v=bfTnAeSinmo&rel=0

Topics in this section

- Create PI AF databases
- View or edit properties of PI AF databases
- Connect to a database on a different PI AF server
- Set the default database
- Rename databases
- Search for PI AF databases
- Refresh the list of databases
- Database deletion

Create PI AF databases

You can create an empty PI AF database with no pre-configured content, or you can load a library that has been saved on a connected PI AF server.

Procedure

1. Choose one of the following actions:

- Press Ctrl+O.
- On the PI System Explorer toolbar, click Database.
- Click **File** > **Database**.

2. From the **Asset server** drop-down list in the Select Database window, select a connected server on which you have administrator access rights to create a database. Alternatively, click ... and select a server from the PI AF Servers window.
3. Click **New Database**.
4. In the Database Properties window, enter a unique name in the **Name** field.
5. Optional. Enter a description in the **Description** field.
6. Choose from the following actions:

To ...	Do this ...
Create a new database with no configured content	Leave the **Load Library** field (if displayed) set to <None>.
Create a database with configured objects from a saved library	Select a library from the **Load Library** drop-down list. For more information, see PI AF saved libraries.
Create extended properties	See Storage of application-specific information.
Set up access permissions for the new database	See Configure security for a single PI AF database.

7. Click **OK**. The new database is added to the **Database** list in the Select Database window.

View or edit properties of PI AF databases

Use the Database Properties window to check how many objects exist in each collection type in a PI AF database, as well as to review or add extended properties and access permissions.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Properties**.
3. In the Database Properties window, choose from the following actions:

To ...	Do this ...
Review the number of objects in the database	Click the **Counts** tab.
Review or modify the database name and description	On the **General** tab, make changes as needed in the **Name** and **Description** fields.
Review or modify extended properties	a. On the **General** tab, click **Extended Properties**. b. In the Extended Properties window, make changes as needed in the **Name** and **Value** fields. To create a new property, click **New Extended Property**. For more information, see Create extended properties.

Review or modify database security	a. On the **General** tab, click **Security**. b. In the Security Configuration window, make changes as needed. For more information, see Configure security for a single PI AF database.
Review the properties of the connected server	a. On the **General** tab, click . b. In the PI AF Server Properties window, review properties on the **General** tab, as described in View general properties of the connected PI AF server.
Load a saved library	a. On the **General** tab, select a saved library from the **Load Library** list. For more information on saved libraries, see PI AF saved libraries. b. Click **Apply**.

4. Click **OK**.

Connect to a database on a different PI AF server

Use the PI AF Servers window to locate and connect to the PI AF server that contains the database you want to work with.

Procedure

1. In the Select Database window, click .
2. In the PI AF Servers window, review the list of servers. Note that the default database on each connected server is also displayed.
3. Choose from the following actions.

To ...	Do this ...
Select the server where the database is located	Double-click on a connected ( ) server. The PI AF Servers window closes and the selected server is displayed in the **Asset server** field.
Check the properties of a server	Select a server and click **Properties**.
Connect to a server	Right-click a server that is not connected ( ) and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server that is not connected and click **Connect As**. b. In the Connect to PI AF Server *server* window, enter a Windows user account name and password. c. Click **OK**.

Review access permissions for a connected server	a. Right-click a connected server and click **Security**. b. In the Security Configuration window, check the settings for the listed identities, as needed. **Note:** If you have administrative privileges on the server, you can also modify the security configuration. See Configure security for new PI AF databases.

Set the default database

The default database is displayed when PI System Explorer opens for the first time. After the first time you run PI System Explorer, it will display whatever database was open when you last closed PI System Explorer.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Set as Default Database**. A check mark icon ( ) is displayed over the database icon that is now the default.
3. Click **OK**.

Rename databases

You can change the name of a PI AF database as needed.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Rename**.
3. Enter a valid name for the database. For rules on object names, see Valid characters in PI AF object names.
4. Click **OK**.

Search for PI AF databases

To locate a specific PI AF database on a selected PI AF server, use the Search feature.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, begin typing the name of a database in the **Databases** field.

   The **Search** icon changes to the **Searching** icon . Results filter corresponding to how many letters you type.

3. Select a database in the list and click **OK**. The window closes, and you will be working in the selected database.
4. To return to the full list of available databases, press ESC. The window closes.

Refresh the list of databases

When other users add databases to the PI AF server, those databases might not be displayed until you refresh.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click in the list of databases and click **Refresh**.

Database deletion

When you delete a database, all information contained within that database is removed. Before you delete a database, therefore, ensure that you have selected the database that you want to delete.

> **Caution:**
> You cannot undo a deletion. However, you can recover data by restoring your last SQL Server backup.

PI AF and PI Data Archive collective connections

An administrator needs to configure connection preferences if a PI AF collective or PI Data Archive collective is being set up.

Topics in this section

- Manage connection preferences for PI System Explorer
- Switch collective members
- Switch to a specific collective member

Manage connection preferences for PI System Explorer

Use the **Server Options** window to define how PI System Explorer should connect to a PI AF collective or PI Data Archive collective, whether login prompts are allowed, and if you want servers to be added to the Known Servers Table automatically.

Procedure

1. In PI System Explorer, choose **Tools** > **Options** > **Server Options**.
2. In the **PI AF Server Connection Settings in PI System Explorer** section, define how PI System Explorer should connect to the PI AF server:

Setting	Description
Connection preference	Use this preference to influence the selection of a collective member when a connection is made to a PI AF collective. ◦ **Prefer Primary** PI System Explorer attempts to connect with the primary server in the collective, and if it is not available, uses the individual priority settings of the collective member to influence selection of the server connection. This is the default setting. ◦ **Require Primary** PI System Explorer is required to connect with the primary server in the collective. If the primary server is not available, the connection fails. ◦ **Any** PI System Explorer can connect with any available member of the collective, and uses the individual priority settings of the collective member to influence selection of the server connection.
Server name resolution behavior	Use this preference to inform PI System Explorer what action to perform when it attempts to communicate with one or more PI AF servers that are not in the Known Servers Table. ◦ **Auto Add** Adds a server to the Known Servers Table automatically if the computer name is valid and PI System Explorer can find the server's computer name on the network. ◦ **Auto Add If Resolvable** Adds a server to the Known Servers Table only when the server name is successfully resolved by a DNS server. ◦ **Do Not Auto Add** Makes no attempt to find a server if it cannot be found in the existing collective.
Allow login prompt	Select this option if you want a login prompt to appear when a connection to the server fails because of a security exception.

3. In the **PI Data Archive Connection Settings in PI System Explorer** section, define how PI System Explorer should connect to PI Data Archive:

Setting	Description
Connection preference	This preference is used to influence the selection of a collective member when a connection is made to a PI Data Archive collective. ◦ **Prefer Primary** PI System Explorer attempts to connect with the primary server in the collective, and if it is not available, uses the individual priority settings of the collective member to influence selection of the server connection. This is the default setting. ◦ **Require Primary** PI System Explorer is required to connect with the primary server in the collective. If the primary server is not available, the connection fails. ◦ **Any** PI System Explorer can connect with any available member of the collective, and uses the individual priority settings of the collective member to influence selection of the server connection.
Server name resolution behavior	Use this preference to inform PI System Explorer what action to perform when it attempts to communicate with one or more PI Data Archive servers that are not in the Known Servers Table. ◦ **Auto Add** Adds a server to the Known Servers Table automatically if the computer name is valid and PI System Explorer can find the server's computer name on the network. ◦ **Auto Add If Resolvable** Adds a server to the Known Servers Table only when the server name is successfully resolved by a DNS server. ◦ **Do Not Auto Add** Makes no attempt to find a server if it cannot be found in the existing collective.

4. In **Protocols** in the **PI Data Archive Connection Settings** section, set the authentication protocols and the order in which they are used when connections are attempted to known servers.

   PI System Explorer attempts to connect using the first protocol listed. If the first attempt fails, it continues attempting to connect using the other protocol in the list (if selected). If attempts using all protocols fail and **Allow login prompt** is selected, the PI Data Archive **Login** window is displayed.

   a. To establish the order used when connecting to a server, use the arrows button to move protocols.

   b. Select the **Allow login prompt** option if you want a login prompt to be displayed when a connection to the server fails because of a security exception.

5. Click **OK**.

Switch collective members

When you connect to a PI AF collective, PI AF automatically connects you to the collective member with the highest priority (lowest number). You can switch the connection to the next member in the collective list. The next member in the list is determined by members' assigned priority.

Procedure

1. In PI System Explorer, click **File** > **Connections**.
2. Right-click the collective and click **Switch Collective Member**.

Switch to a specific collective member

When you connect to a PI AF collective, you are automatically connected to the collective member with the highest priority (lowest number). You can switch to a specific member of the collective.

Procedure

1. In PI System Explorer, select **File** > **Connections**.
2. Right-click the collective and click **Connect to Collective Member**.
3. In the Choose Collective Member window, select the collective member from the **Collective Member** list to which you want to connect.
4. Click **OK**. You are now connected to the selected collective member.

Database import and export

You can export a database from PI System Explorer in XML and then restore it from that XML file by importing it back to PI AF.

To export the database without having to manage XML files, you export the database as a library, as described in Save databases as libraries.

Videos

For information on how to import and export databases with PI System Explorer, watch this video:

https://www.youtube.com/watch?v=ClGTy_lkoTg&rel=0

For information on how to import and export databases with import and export utilities, watch this video:

https://www.youtube.com/watch?v=zc5pcH1XvMo&rel=0

Topics in this section

- Export a database to XML
- Restore databases from exported XML files
- Export of library objects to another database
- AFExport utility
- AFImport utility

Export a database to XML

This procedure archives PI AF databases into an XML file that you can later restore.

> **Note:**
> You can also export PI AF objects from a command line utility. See AFExport utility.

Procedure

1. Click **File** > **Export to File**.
2. In the Export to File window, select desired export options (XML export options).
3. Click **OK**.

   PI System Explorer exports the current database into an XML file.

Topics in this section

- XML export options
- All Referenced Objects
- File format for export and restore

XML export options

The Export to File window contains the following export options.

Export option	Result
Include All Referenced Objects	Causes dependent references to be exported, as detailed in the All Referenced Objects table. Use this option to facilitate moving objects between different PI AF databases.
Include Security Settings	Causes the security settings of individual objects to be exported. This option increases the amount of time required for the export and subsequent import operations. You must have Administrative privileges to import objects that have the security setting specified.
Flatten XML	Exports hierarchical objects in a flat format. Exporting in a flat style can make editing in some tools simpler. Hierarchical objects that will be exported flat are attributes, attribute templates, elements, and event frames.
Simplify Configuration Strings	Removes PI Data Archive and tag identifiers from configuration strings for PI point data references. Additionally, substitution parameters will be resolved. **Note:** This option slows the export process because it requires evaluation of the saved configuration strings.
Include Default Values	Includes the default values assigned to objects. Without this option, a property that has not been changed from its default setting is not included in the output. This saves considerable space and time when you are exporting large amounts of data.
Include Unique IDs	Includes the unique ID of each object in the export. By including this option, you can rename existing items during an import to the same database. Unless rename is required, it is more efficient to leave this option turned off.
Library Objects Only	Disables the event frames, transfers, and cases options.
Include Event Frames, Transfers, and Cases	Exports event frames, transfers, and cases.

All Referenced Objects

The following table lists the objects that are exported when you select **Include All Referenced Objects**.

XML object type	Included reference
`AFAnalysis`	`AFAnalysisCategories` referenced
	`AFAnalysisTemplate` in `Template`
	`AFElementTemplate` in `CaseTemplate`
	`AFElement` in `Target`
	`AFSecurityIdentities` from `SecurityAccessControl` (if **Include Security Settings** is also selected)

XML object type	Included reference
AFAnalysisTemplate	AFAnalysisCategories referenced
	AFElementTemplate in CaseTemplate
	AFElement in Target
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFAttribute	Referenced AFEnumerationSets in ValueTypeQualifier when not from template
AFAttributeTemplate	Referenced AFEnumerationSets in ValueTypeQualifier
AFCase	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	All child AFElements
	All AFConnections (not just those added or removed)
AFContact	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFDatabase	UOMDatabase
	AFContacts referenced
	AFNotificationContactTemplates referenced
	AFSecurityIdentities (if **Include Security Settings** is also selected)
AFElement and AFModel	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFReferenceTypes to referenced children
	Same as above for all child elements
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFElementTemplate	Base AFElementTemplates
	AFElementTemplates referenced in AFPort.AllowedElementTemplate
	AFReferenceTypes that specifically reference this template
	AFElementCategories referenced
	AFEnumerationSets referenced in attribute templates
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFEnumerationSet	AllReferences adds AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFEventFrame	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFReferenceType from parents
	AFElements referenced
	Same as above for all child event frames
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)

XML object type	Included reference
AFModelAnalysis	AFAnalysisCategories referenced
	AFElementTemplate in Template
	AFElementTemplate in CaseTemplate
	AFElement in Target
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotificationTemplate	AFNotificationContacts
	AFAnalysisTemplate
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotification	AFNotificationTemplate in Template
	AFNotificationContacts
	AFAnalysis
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotificationContact	AFNotificationContactTemplate
	Child AFNoticationContacts
	AFContact referenced
AFNotificationContactTemplate	Child AFNotificationContactTemplates
	AFContacts referenced
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFDeliveryFormat	DeliveryChannel
AFNotificationRule	AFNotificationRuleTemplate
	AFNotificationContactTemplate (via Subscribers)
	Child AFNoticationContacts
	AFContact
	AFDeliveryFormat
AFNotificationRuleSubscriber	AFNotificationContactTemplate
	Child AFNoticationContacts
	AFContact
AFNotificationRuleTemplate	AFNotificationContactTemplate (via Subscribers)
	Child AFNoticationContacts
	AFContact
	AFDeliveryFormat
AFReferenceType	AFReferenceTypeCategories referenced
	AFElementTemplates from AllowedParentElementTemplate and AllowedChildElementTemplate
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFSecurityIndentity	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)

XML object type	Included reference
AFSecurityMapping	AFSecurityIdentity mapped to
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTable	AFTableConnection
	AFTableCategories referenced
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTableConnection	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTransfer	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFElements referenced in Source and Destination
UOMClass	UOMClass for exported class and classes referenced by UOMs in UOMClass. For example, when the Area class is exported, the Length class is included because Length UOMs are referenced by many Area UOMs. **Note:** UOMClass export does not include security settings, which are only exported with UOMDatabase.
	UOMGroup (if any of the UOMs in the exported or referenced classes have an assigned UOMGroup).
	UOM for all UOMs belonging to the exported UOMClass.
UOMDatabase	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
	UOMGroup. Group mappings are defined under each UOM.
	UOMClass
	UOM

File format for export and restore

The Import/Export XML format is described in a schema file located in the PIPC\AF installation directory, OSIsoft.AF.xsd.

Many of the AF Objects support an operation attribute in the XML that allows for deletion. Similarly, you can execute the **Auto Point Config** option selectively on elements using this same technique. Example:

```
...
<AFElement operation=delete> <Name>ElementToDelete</Name> </AFElement>
...
```

Elements and Attributes can be imported using a flat list, in which the relative path of the element or attribute is included in the name field. Example:

```
...
<AFElement> <Name>RootElement</Name> ... </AFElement>
<AFElement> <Name>RootElement\ChildElement1<\Name> ... </AFElement>
```

```
<AFElement> <Name>RootElement\ChildElement2<\Name> ... </AFElement>
...
```

Restore databases from exported XML files

You can restore a database that has been exported to XML by importing objects from the XML file.

You can also import PI AF objects from a command line utility. See AFImport utility.

Before you start

OSIsoft recommends that you do not use the **Allow Update** option when importing objects with duplicate names, such as event frames, into an empty PI AF database. Otherwise, problems can occur. However, you may then receive a message that objects, such as units of measure, already exist on the PI AF server. The import does not attempt to make any changes to existing objects, which means that if there are differences between objects' properties in the XML file and the destination system, they are not resolved.

Procedure

1. Click **File** > **Import from File**.
2. In the Import from File window, select the XML file that contains the data and select the appropriate options.

Import option	Description
Allow Create	Allows new objects to be created. If you want to update existing items, clear this option to prevent accidental creation of new objects.
Allow Update	Allows existing objects to be updated. If you only want to add new items, clear this option to prevent existing objects from being accidentally overwritten.
Automatic Check In	Allows the import operation to automatically check in objects as the import proceeds. This reduces the maximum memory requirements of the operation. Use this option when you are importing a large number of objects.
Create or Update PI Points	Causes any attributes with a PI point configuration specified in the XML file to be updated if they already exist, or created if they do not exist. This option invokes the appropriate Data Reference **CreateConfig** option, which creates or updates PI points, as well as resolving their substitution parameters and setting server and point IDs. The performance of the import operation is negatively affected when this option is selected. **Note:** PI points are not created unless the **Tag Creation** check box is selected in the PI Point Data Reference window.

Preserve Unique IDs	Retains any unique IDs that exist in the source XML. This option is only valid when the **Allow Create** and **Automatic Check In** options are also selected. The performance of the import operation is negatively affected when this option is selected. There is no guarantee that PI AF will preserve unique IDs of attributes derived from templates because unique IDs of such attributes are generated based on the template's unique ID. **Note:** Because PI AF servers maintain a cache of `rowids` to `ids`, the import with **Preserve Unique IDs** option affects load-balanced PI AF servers differently from non-load balanced PI AF servers. In a non-load balanced scenario, removed items will cause the cache of `rowids` to `ids` to be updated. However, in a load-balanced scenario, the non-connected server will need to be restarted to have the cache updated after items are removed, and before the import with **Preserve Unique IDs** option is used to restore items to the same system.

Export of library objects to another database

You can export library objects (templates, formulas, UOM, and so on) from one database into another.

Topics in this section

- PI AF saved libraries
- Save databases as libraries
- Load database libraries
- Review properties of loaded libraries

PI AF saved libraries

A PI AF saved library provides a collection of application- or domain-specific objects that you can import into a PI AF database. Saved libraries typically include categories, element templates, enumeration sets, reference types, tables, as well as the unit-of-measure database, which is always included. You also have the option to include other objects, such as elements and notifications.

Note:
Libraries are stored as XML files in the PI AF SQL Server database (PIFD), where they are easily accessible to other users. By contrast, the **Export to File** option enables you to export an entire database of objects as an XML file to your computer or network. The exported file can be imported to a different server altogether.

Save databases as libraries

You save database objects as a library when you wish to make those objects available for import to a different database on the PI AF server.

Procedure

1. Click **File** > **Save as Library**.
2. In the Save Database as Library window, enter a name in the **Library Name** field. The library name does not need to be unique if you plan to overwrite an existing library.
3. Optional. Enter a description of the library content in the **Description** field.
4. To replace an existing library, select the **Replace existing Library without prompting** check box.
5. To include objects such as elements and notifications in the library, select the **Include non-library objects** check box.
6. Click **OK**.

Load database libraries

You load a database library from a set of libraries that have been previously saved in the PI AF database.

Procedure

1. Click Database.
2. In the Select Database window, select the database into which you want to load a library, and click **OK**.
3. Click **File** > **Load Library**.
4. In the Load Library into Database window, select a library from the list and click **OK**.

Results

The objects in the loaded library populate the current database. To confirm how many objects are loaded, view the **Count** tab in the Database Properties window, as described in View or edit properties of PI AF databases.

Review properties of loaded libraries

You view the properties of libraries that have been loaded on the PI AF server in the PI AF Server Properties window. You can change a library name and description.

Procedure

1. Click **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Libraries** tab.

   The libraries that are currently installed are listed.
3. Right-click the library you want to review and click **Properties**.

4. Optional. In the Library Properties window, you can change the library name and description as needed.
5. Click **OK**.

AFExport utility

The AFExport utility is a command line application that you can use to archive PI AF databases into an XML format that you can restore later. Use this utility to archive elements, templates, event frames, transfers, and other objects from a PI AF database. You can also export collections of PI AF objects from the PI System Explorer. See Export a database to XML.

The `AFExport.exe` utility is located in the `\PIPC\AF` folder.

To run the PI AF Export utility you open a command window and navigate to the `PIPC\AF` folder. Use the following syntax: `AFExport.exe` and choose from the parameters listed in AFExport utility parameters. To display all parameters, type:

```
AFExport /?
```

Topics in this section

- Guidelines for exporting collections
- AFExport utility parameters
- Sample export paths

Guidelines for exporting collections

You should follow these guidelines as you prepare to export a collection:

- Specify collections by their PI AF SDK property name, followed by "[ ]".
- Identify individual members of a collection by placing their name within the brackets.
- If no collection name is specified, the default child collection for that location in the path is assumed.
- The default child collection of a PI AF server is `Databases`.
- The default child collection of a database is `Elements`.
- The default child collection of an element is `Elements`.
- The default child collection of UOM database is `UOM`.
- A vertical bar (|) indicates an attributes or attribute templates collection.

AFExport utility parameters

The **`AFExport`** utility supports the following parameters.

Parameter	Short form	Description
Path Additional details and examples follow this table.		Path to the object that you wish to export. Typically of the form: *****afserver* *****database*. Use '`.`' to export the default database. See Sample export paths.

Parameter	Short form	Description
/File:*string*	**/F**	Specify an output file. If not specified, then the output is streamed to the console.
/StartTime:*string*	**/T**	Specify a start time to cause event frames, transfers, and cases to be exported that are between the start and end times specified.
/EndTime:*string*	**/E**	Specify an end time when exporting event frames, transfers, and cases.
/AllReferences[-]	**/A**	Export any referenced object from the specified PI AF object. This parameter cannot be used with the **No References** parameter.
/NoReferences[-]	**/N**	Do not export any referenced objects, including child objects. This parameter cannot be used with the **All References** parameter.
/DefaultValues[-]	**/D**	When this parameter is not specified, then properties that are set to the default values are not exported, resulting in a substantially smaller export file. Importing a file without default values over existing objects may result in values not being reset to their defaults.
/Library[-]	**/L**	Export only the library objects from a database. Library objects include all categories, templates, enumeration sets, reference types, and tables.
/Security[-]	**/Y**	Export security information. Exporting security information will increase both export time and subsequent import times.
/SimplifiedConfigStrings[-]	**/Sc**	Export configuration strings for attribute templates or attributes derived from attribute templates without substituting parameters and do not export UniqueIDs or point identifiers in the configuration strings.
/UniqueIDs[-]	**/U**	Export unique IDs of all objects. This allows the renaming of objects when imported back into the same database. This parameter increases the size of the output file and may increase the amount of time required to import.
/Silent[-]	**/S**	Silent mode. Prevents informational messages from being displayed. If no output file is specified, this option is automatically chosen.

Parameter	Short form	Description
/Summary[-]	**/M**	Summary mode. Output only minimal information on progress. This parameter is not valid with the **Silent** mode or if no output file is specified.
/User:*string*	**/user**	Use to specify a different Windows user account to connect to the PI AF server.
/Password:*string*	**/pw**	If a user name is specified, specify the network credentials password.
/Version	**/V**	Use to display version information. All other parameters are ignored.
/Flat[-]	**/flat**	Export hierarchical object in a flat format. Exporting in a flat style can make editing in some tools simpler. Hierarchical objects that will be exported flat are attributes, attribute templates, elements, and event frames.
/Paste[-]	**/Pa**	Paste operation behaves as a typical copy/paste.
/?, **/help**		Prints the contents of this table.
@*file*		Use the specified file to provide additional input arguments. The file should contain one argument per line. Comment lines start with the **#** character.

Sample export paths

The following table contains sample syntax to export different PI AF database components.

To export ...	Use this syntax ...
The default database	`.`
The database `MyDatabase` on the default PI AF server	`\\.\MyDatabase`
The element `MyElement` in the database `MyDatabase` on the PI AF server named `MyAFServer`	`\\MyAFServer\MyDatabase\MyElement`
All element templates in the database `MyDatabase`	`\\.\MyDatabase\ElementTemplates[]`
Element template `T1` in the database `MyDatabase`	`\\.\MyDatabase\ElementTemplates[T1]`
All enumeration sets in the `MyAFServer` default database	`\\MyAFServer\.\EnumerationSets[]`
All attributes of `MyChildElement` in the `MyAFServer` default database	`\\MyAFServer\MyElement\MyChildElement\Attributes[]`
The UOM database (`UOMDatabase` is case sensitive)	`\\.\UOMDatabase`
All tables in the `MyAFServer` default database	`\\MyAFServer\.\Tables[]`

AFImport utility

The **AFImport** utility is a command line application that you can use to restore PI AF objects into a database. You can also use **Import from File** in PI System Explorer to restore database objects. See Restore databases from exported XML files.

The **AFImport.exe** utility is located in the \PIPC\AF folder.

To run the **AFImport** utility, you open a command window and navigate to the PIPC\AF folder. Use the following syntax: AFImport.exe and choose from the parameters listed in AFImport utility parameters. To display all parameters, type:

```
AFImport /?
```

AFImport utility parameters

The **AFImport** utility supports the following parameters.

Parameter	Short form	Description
Path		Path to the object into which you want to import. Typically of the form: *****afserver**database*. Use '.' to import into the default database.
/File:*string*	**/F**	Specify an input file. If not specified, the import operation reads from the standard input.
/AutoCheckIn[-]	**/A**	Automatically check in changes during an import operation. Default value: True
/Create[-]	**/C**	Allow import operation to create new objects. Default value: True
/Update[-]	**/U**	Allow import operation to update existing objects. Default value: True
/CreateCategories[-]	**/CC**	Create categories that do not exist but are referenced by objects in the input. Default value: True

Parameter	Short form	Description
/PreserveUniqueIDs[-]	**/Pid**	Retain unique IDs that exist in the source XML. This option is only valid when **/Create** and **/AutoCheckIn** are also specified. This option can significantly affect import performance. There is no guarantee that PI AF will preserve unique IDs of attributes derived from templates because unique IDs of such attributes are generated based on the template's unique ID. **Note:** Because PI AF servers maintain a cache of `rowids` to `ids`, the import with **Preserve Unique IDs** option affects load-balanced PI AF servers differently from non-load balanced PI AF servers. In a non-load balanced scenario, removed items will cause the cache of `rowids` to `ids` to be updated. However, in a load-balanced scenario, the non-connected server will need to be restarted to have the cache updated after items are removed, and before the import with **Preserve Unique IDs** option is used to restore items to the same system.
/CreateUpdatePIPoints[-]	**/P**	Create or update PI point configuration for newly created elements. This option can significantly affect import performance.
/DisableConfigStringValidation[-]	**/D**	Disable the validation of configuration string settings for data references and delivery channels, which can improve the speed of the import operation. However, this will bypass looking up PI points, which corrects or adds server IDs and point IDs to configuration strings.
/GenerateUniqueNames[-]	**/G**	Generate unique names for objects when an object with the same name already exists.
/Paste[-]	**/Pa**	Paste operation behaves as if this is a copy/paste process. Depending on where the data is imported, this affects whether a new name is generated.

Parameter	Short form	Description
/ContinueOnError	**/CE**	Specify the behavior of the import process when a recoverable error occurs. The three options available are `Yes`, `No`, and `Prompt`. The default value is `Prompt`, unless running silently in which case the default will be to cancel the import process.
/Silent[-]	**/S**	Silent mode. Prevent informational messages from being displayed.
/Summary[-]	**/M**	Summary mode. Output only minimal information on progress.
/User:*string*	**/user**	Use to specify a different Windows user account to connect to the PI AF server.
/Password:*string*	**/pw**	If a user name is specified, specify the network credentials password.
/Version	**/V**	Display version information. All other parameters are ignored.
@*file*		Read response file for more options. The response file contains one parameter per line. Comment lines start with the **#** character.

Retrieval of asset information

PI System Explorer provides a variety of methods that you can use to locate and review the assets in your PI AF database, including a browser, search windows, trends to verify formulas and calculations, and version management. You can also review time series data for attributes that contain or reference such data.

Topics in this section

- Asset browsing
- Asset and asset data searches
- Trends in PI System Explorer
- Version management for equipment and processes
- Display of different object versions and obsolete objects
- View time series data

Asset browsing

You use the PI System Explorer browser to display asset objects in the PI AF database. The browser displays the following assets, depending on the asset type that you have selected in the navigator pane:

- **Elements**

  Displays element assets and element searches in a tree format. You can control how many elements are displayed per page.

- **Event Frames**

  Displays event frame and transfer search collections. Because asset models can comprise thousands of event frames or transfers, these objects are not displayed in a hierarchy in the browser. Instead, each has a search collection in the browser. Recent searches are also listed under the search collection.

- **Library**

  Displays object collections for **Templates** (for elements, event frames, models, and transfers), **Enumeration Sets**, **Reference Types**, **Tables**, **Table Connections**, and **Categories** (for analyses, attributes, elements, notification rules, reference types, and tables). To see all the objects of a particular type, you expand the collection for that type.

- **Unit of Measure**

  Displays all UOM classes in the Unit of Measure database. When you select a class, UOMs belonging to that class are listed in both the viewer and the conversion calculator.

- **Contacts**

  Displays a list of contact searches, escalation teams, groups, and stand-alone delivery endpoints for use with notifications.

Topics in this section

- Element browsing

- Configure browser page size
- Open additional browser windows

Element browsing

Elements are displayed as a tree in the browser. The structure of the element tree is different for each organization. The asset model designer chooses structure that is relevant to the users in the organization. In the following illustration elements are organized by plant location. If you were in charge of all the equipment for a particular plant, then such a model might make sense for you. A different model might be more useful for someone with a different role.

The element tree can include different sub trees that provide different context for the same assets. This allows users to find elements in the context that makes the most sense for the task at hand. For example, in addition to the organization by plant location shown above, you might also have an organization by equipment manufacturer, as shown in the following illustration.

The elements representing the pumps appear in both the Manufacturers and the Plant Location hierarchies. In the Manufacturer hierarchy, the pump elements are not new separate elements, rather they are references to the elements that already existed in the Plant Location tree. To indicate that they are element references, they are represented by the referenced element icon .

Configure browser page size

The PI System Explorer browser is paged. This means that it displays a certain number of objects at a time. If more objects exist, PI System Explorer displays them in another page. When this happens, you see <Previous> and <Next> in the browser.

By default, the maximum number of objects displayed in the browser is 1000. You can change that number.

Procedure

1. In PI System Explorer, select **Tools** > **Options**.
2. On the **General** tab, type the maximum number of objects that can be displayed in the browser in the **Maximum Tree Branch Page Size** field.
3. Click **OK**.

Open additional browser windows

Occasionally, you may find it convenient to open additional browser windows. For example, you can have the Unit of Measure database displayed in one browser instance while creating a new template in the Library browser.

Procedure

1. In the navigator pane, right-click the item you want to open in another window.
2. Select **Open in New Window**.
3. To move between windows, click the PI System Explorer icon in the Windows taskbar and select the thumbnail of the window you want.

Asset and asset data searches

PI System Explorer provides a variety of options for finding assets, asset data, and PI points.

- Find data for a specific piece of equipment. For example: PI point data, calculation results, maintenance information, and so on. For example, you can use PI System Explorer to find information about a specific tank at a specific plant.
- Find all equipment with specified attribute values or value ranges. For example, get a list of all tanks with temperature greater than 200 °F.

Topics in this section

- Quick search
- Search result paging
- Maximum query sizes
- Searches on a specified date
- Search for PI points
- Manage search results
- Search for elements
- Special characters in name searches
  You can use special characters when searching for objects by name.
- Search for attributes on elements
- Search for event frames

- Search for attributes on event frames
- Search for transfers
- Search for attributes on transfers

Quick search

You can use the quick search box at the right end of the PI System Explorer toolbar to find elements, event frames, element templates, or UOM (unit of measure) types.

Procedure

1. Make a selection in the navigator pane to set the context for quick search. For example, select **Event Frames** to set quick search to search for event frames.
2. In the quick search box, select a search constraint from **Actions** ▾, enter search criteria, and press Enter. Search results are displayed in the viewer, as well as in the browser.

> **Note:**
> Search results that are displayed in the browser remain there for easy access in the event you need them again. You can right-click and delete them from the browser; they are also removed when you exit PI System Explorer. You can right-click and save them to keep them until you delete them.

Search result paging

When you are searching large numbers of elements or event frames, you can use the **Set Maximum Query Size** option to return quick search results back in manageable chunks. This is called *paging*. Change the **Set Maximum Query Size** value to the number of objects you want to see in the search results at one time. If the search returns a greater number of objects than the **Set Maximum Query Size** value specifies, the results are paged.

For example, if the **Set Maximum Query Size** value is 100,000 the PI AF server attempts to find 100,000 matching objects before returning the results. You probably would not want to wait for the server to find that many matches, nor would you want to see that many matches in a single page of search results. If you change **Set Maximum Query Size** to 100, the PI AF server would send you the results after each 100 matches, and you would see 100 objects at a time in the search results.

Search result rules

- Only search results for elements, transfers, and event frames are paged.
- The **Set Maximum Query Size** setting applies only to quick searches of elements, transfers, or event frames. If you are searching other objects, the search ignores the **Set Maximum Query Size** value.
- In search windows, the **Results per Page** value overrides the **Set Maximum Query Size** value.

Maximum query sizes

A system default controls the number of objects that can be returned by a search query. Users can override the system default by specifying a different value in Element Search, Attribute Search, Event Frame Search, and Transfer Search windows.

System default

You set the **Maximum Query Size** option in **Tools** > **Options**. This sets the default value for PI System Explorer.

Custom sizes

You can override the system default for a specific search by setting a custom value for **Maximum results**. For event frames, you specify a custom value in the **Results per Page** field.

Searches on a specified date

You can conduct a search in PI System Explorer at a specified time and date, by entering that time setting for **Query Date**. PI System Explorer also displays object versions for that date. See Query Date and PI System Explorer time for information on what objects are affected by **Query Date**.

In addition, you can set a time or time range that applies only to displayed attribute values and leaves in place the **Query Date** setting. To do this, establish a time context in the Set Time Context window, as described in Set time context for displayed attribute values.

Search for PI points

Use the Tag Search window to retrieve PI points that match your search query criteria. In PI AF and PI Builder 2017 or later versions, you can create a search query based on both PI point attributes and PI point values.

> **Tip:**
> To create PI point data references directly from PI points retrieved from a PI point search, select **View** > **Palette** > **Tag Search**. For more information, see Create direct PI point data references from Tag Search results.

Procedure

1. In the navigator pane, click **Elements**.
2. Select **Search** > **Tag Search**.
3. In the Tag Search window, select the PI Data Archive computers on which you want to search.

   To add or change PI Data Archive computers, choose from the following actions:

- Click ▼ next to the **Server(s)** field and click a PI Data Archive on the list.
- Click ... and double-click on the PI Data Archive you wish to add from the list in the PI Data Archives window.
- To remove a selected PI Data Archive, select the server name in the **Server(s)** field, right-click and click **Delete**.

Servers are added in alphabetical order to the existing selection. You can add multiple selections from either list.

4. You create a PI point search query in the text box next to the **Search** button. A search query comprises a filter name, which can be a text string, a PI point attribute (defaults to **`tag`**) or a PI point value (for example, **`Value`**), an operator (defaults to `:=`), and a query filter (defaults to *, or all PI points). For a full description of the syntax of a PI point query, see Syntax for PI point searches.

   Examples of attribute queries are:

   - `sin*`
   - `tag:=sin*`
   - `tag:<>sin* datatype:float`
   - `step:0 AND pointsource:L`
   - `tag:<>sin* AND PointType:Float64`

   Examples of value queries (which you can combine with attribute queries) are:

   - `Value:>1000`
   - `tag:<>sin* AND Value:>10`
   - `PointType:Int32 AND Value:>10`
   - `PointSource:L AND Annotated:1 AND TimeStamp:t`
   - `creationdate:>y-1d AND future:true AND timestamp:<*`

   By default, PI AF searches for point names that start with the query string (meaning that the **Starts With** filter is already selected).

   Note:
   Queries that contain a query filter name (such as `name:*sine*`), search tag attributes only. In previous versions of PI AF, queries that contained a query filter name would search the descriptor attribute as well as the specified query filter, unless the descriptor attribute was actually specified as part of the query filter.

   To build a search query, choose from the following actions.

To ...	Do this ...
Clear previously selected criteria	Click **Reset**. All queries are cleared, as well as the **Include Description in Search** filter.

Include point descriptions in a query	a. Click . b. Click **Add Criteria** or press ALT+C. c. Select **Include Description in Search** and set the value to **True**.
Type a query directly in the search text box	a. Type the first character of the query. b. Select from the list of matching point attributes and values, or continue typing a point attribute or value manually. c. Enter the criteria to be matched.
Include common PI point attributes in a query	a. Click . b. Enter criteria in any of the following fields: ▪ **Name** (alias for **`tag`** attribute) ▪ **Point Source** ▪ **Data Type** (alias for **`pointtype`** attribute) ▪ **Point Class** (alias for **`ptclassname`** attribute) c. To enter criteria for engineering units and description, click **Add Criteria** or press ALT +C. Then click **Engineering Units** and/or **Description**. Criteria are appended to the string in the search text box, separated by a space. **Note:** The search query returns only PI points that match all criteria.
Include PI point values in a query	a. Click . b. Click **Add Criteria** or press ALT+C. c. Click any of the following criteria: ▪ **Value** ▪ **TimeStamp** ▪ **IsGood** ▪ **Annotated** ▪ **Substituted** ▪ **Questionable** d. Enter criteria in the fields you have selected, as required. Note that you can select a comparison operator for **Value** and **TimeStamp** criteria. Criteria are appended to the string in the search query text box, separated by a space. **Note:** The search query returns only PI points that match all criteria.
Clear an existing query, or remove a criterion from the expanded search area	Click ✕.

Use a previous query	a. Click ▼. b. Click **Recent Searches**. c. Select the existing query from the list.
Filter point names by contents, exact match, or ending	a. Click ▼. b. Click the appropriate filter: ▪ **Contains** ▪ **Exact Match** ▪ **Ends With**

5. Click **Search** to retrieve the points that match into the results table.
6. Choose from the following actions.

To ...	Do this ...
Review time series data	Right-click a PI point and click **Time Series Data**. For more information, see View time series data.
Create a trend	Right-click a PI point and click **Trend**. For more information, see Create trends.
Add a PI point to an existing trend	Right-click a PI point and click **Add to Trend**.
Review current PI point values, time stamps, and attributes	Right-click a PI point and click **Properties**.
Create a PI point data reference	For search results displayed in the palette only. Drag a PI point onto the **Attributes** grid. For more information, see Create direct PI point data references from Tag Search results.
Change the columns in the results table	Click ⚙ and clear the columns that you want to remove.
Add attribute columns to the results table	a. Right-click a column heading and click **Add Column**. b. In the Select Point Attributes window, select the point attributes to be added. c. Click **OK**.
Restore original columns in the results table	Right-click a column heading and click **Restore Original Columns**.

7. Click **OK**.

   The results are displayed in the viewer. Each search is temporarily listed in the browser with a **Tag Search Results** label. To differentiate between searches, you can change a label by right-clicking and clicking **Rename**.

Syntax for PI point searches

Refer to the following sections for details on the syntax for building PI point queries in PI AF and PI Builder. For complete details on PI point query syntax, see "PIPoint Query Syntax Overview" in PI System Explorer **Help** > **AF SDK Reference** > **Overview**, or go to the Tech Support page PIPoint Query Syntax Overview (https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/b8fbb6da-7a4b-4570-a09d-7f2b85ed204d.htm).

Condition filters

To build a PI point query, enter one or more **AND** condition filters that you can also combine with an **OR** condition as needed. Each **AND** condition contains one or more queries, separated by a space or AND. A query consists of a query filter name, an operator, and the query filter. This enables you to specify multiple conditions with a single query, as shown in the following example:

```
(tag:<>sin* AND PointType:Float64) OR (tag:="*Tank*" AND DataType:=Int32)
```

> **Note:**
> You can only use parentheses between **OR** conditions.

You can only reference a filter name once per **AND** condition of the query string. For example, `PointId:>5 AND PointId:<10` generates an error, whereas `PointType:=Int32 OR PointType:=Float32` is valid.

For maximum efficiency, build your query so that you eliminate most items from the retrieved results with your first condition filters.

Query filter names

When querying based on PI point attributes, the query filter name is a PI point attribute name or alias. Common aliases are:

Alias name	Attribute name
Name	Tag
DataType	PointType
Description	Descriptor
PointClass	PtClassName

Starting in PI AF 2017, you can query based on values, in addition to querying PI points based on attribute. However, you cannot use the **OR** condition to query a PI point value. For example, you would generate an error if you were to enter the following queries:

- `IsGood:false OR Annotated:true`
- `PointType:Float AND Value:>10` because `PointType:Float` is implicitly translated to `'PointType:=Float16 OR PointType:=Float32 OR PointType:=Float64'`
- `PointType:Int AND Value:>10` because `PointType:Int` is implicitly translated to `'PointType:=Int16 OR PointType:=Int32'`
- `sin* AND Value:>10` because `sin*` is implicitly translated to `'tag:=sin* OR Descriptor:=sin*'` if the default filter setting for **Include Description in Search** is selected. To be valid, you would need to clear the **Include Description in Search** filter.

Wildcard characters

You can use the following special characters in a PI point query.

> **Note:**
> Results of the examples below assume you are using the default search option, which searches for PI point names that start with your search string.

Special character	Description	Example
*	Substitute any number of unspecified characters	`sin*` Returns all PI points that have names starting with "sin", for example, `sinusoid` and `sinusoidu`.
?	Substitute a single unspecified character	`CD?158` Returns all PI points that have names starting with "CD", followed by any single character, followed by "158" (for example, `CD1158`, `CDA158`, and so on).

`:` or `:=`	When searching for all PI points with a specific attribute value (other than name), separates the attribute and the value you are searching for.
	Tip: When searching for a PI point name that contains a colon, enclose the name in double quotation marks, or precede the colon with a backslash.
		`pointsource:R`
		Returns all PI points that have the **`pointsource`** value `R`.
		`"ba:temp.1"`
		`ba\:temp.1`
		Either of the above examples returns the PI point named `ba:temp.1`.

`''`
or
`""`
	Delimiters for search strings containing spaces or special characters	`'*Owner Change*'`
		or
		`"*Owner Change*"`
		Returns all PI points that have names containing Owner Change.
		`"ba:temp.?"`
		Returns all PI points that have names starting with `ba:temp.` and ending with any single character.

Operators

The following table lists the operators that you can use in an **AND** condition.

Operator	Description	Example
`=`	The **`EQUALS`** operator.	`Tag:Tank*` or `Tag:=Tank*`
`<>`	The **`NOT EQUALS`** operator.	`PointType:<>Int32`
`<`	The **`LESS THAN`** operator.	`Descriptor:<M`
`<=`	The **`LESS THAN OR EQUAL`** operator.	`Tag:<=Tank`
`>`	The **`GREATER THAN`** operator.	`Tag:>Tank`
`>=`	The **`GREATER THAN OR EQUAL`** operator.	`Tag:>=Tank`

In PI point value queries with a String data type, you cannot use the following operators: `<`, `<=`, `>`, or `>=`. Furthermore, when boolean values are expected (as with **`Substituted`**,

`Questionable`, **`Annotated`**, and **`IsGood`** point value queries), you can only use the = and <> operators.

Syntax restrictions

- You cannot query future point attributes, such as `creationdate:>y-1d AND future:true`, on PI Data Archive servers older than 3.4.395.
- You cannot query security point attributes, such as **`PtSecurity`** and **`DataSecurity`**, on PI Data Archive servers older than 3.4.380.

Manage search results

PI System Explorer displays all search results in the browser where you can save them in a searches collection for reuse, rename them, or edit the criteria. All unsaved search results are indicated by an asterisk.

Note:
All unsaved search results are deleted when you exit PI System Explorer.

Procedure

1. Optional. To work from a list of search results in the viewer rather than in the browser, choose from the following actions:

To ...	Do this ...
Manage an element or attribute search	a. In the navigator, click **Elements**. b. In the browser, click the **Element Searches** collection.
Manage an event frame search	a. In the navigator, click **Event Frames**. b. In the browser, click the **Event Frame Searches** collection.
Manage a transfer search	a. In the navigator, click **Event Frames**. b. In the browser, click the **Transfer Searches** collection.

2. Select a search and choose from the following actions:

To ...	Do this ...
Save a search result	Right-click an unsaved search result name and click **Save**.
Rename a search result	a. Right-click a search result name and click **Rename**. b. Type a new search result name.
Copy a search result	Right-click the search result name and click **Copy**.
Edit criteria for a search result	a. Right-click a search result name and click **Properties**. (In the viewer, you can also double-click a search result name.) b. Modify search criteria as needed and click **OK**.

Delete a search result	a. Right-click a search result name and click **Delete**. b. Click **Yes** in the Delete confirmation window.
Rearrange event frame or transfer search results	In the browser only: a. Right-click an event frame or transfer search result name and click **Arrange By**. b. Select from the following options: ▪ **Arrange By Name** ▪ **Arrange By End Time** ▪ **Arrange By Start Time** ▪ **Arrange By Category** ▪ **Arrange By Template**
Display the full path to elements or event frames	In the viewer only: a. Right-click on the column header (or below the search results grid). b. Click **Show Full Paths**.

Search for elements

Use the Element Search window to retrieve element data that matches your search criteria.

Procedure

1. In the navigator pane, click **Elements**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Element Search**.
   - In the browser, right-click the **Element Searches** root node and select **New Search**.
3. Set the Element Search window to find the desired element or elements in the PI AF database.

   Note:
   If you enter values for multiple criteria, the search returns only those elements that match all the specified criteria.

4. From the **Actions** ▾ list, select the type of filter to apply: **Contains**, **Exact Match**, **Starts With**, or **Ends With**.
5. Choose from the following actions:

To ...	Do this ...
Type one or more filter conditions directly into the **Enter element criteria** field	◦ Use the : or := operator to select elements that match a name, description, template, and category. ◦ Use special characters as needed. For more information, see Special characters in name searches. ◦ Enclose strings that contain spaces or special characters with the " character. ◦ Separate filter conditions with a whitespace, for example: `Category:"Processing Plants" Name:Cracking`

Filter elements under **Criteria**	Enter values in the following fields as needed. Click **Add Criteria** to enter criteria for additional fields.

- **Name**

  Enter the name of the element to retrieve, based on the filter type. You can enter special characters to match part of a name. See Special characters in name searches.

- **Element Search Root**

  Enter the element that you want to use as the root or base level for the element search. Type the exact name or click ... to open the Element Browser window, where you can view the element hierarchy and select an element. You cannot include wildcard characters in the entered name. If you do not specify an element, you set the main level of the element hierarchy as the root.

  Depending on your PI AF hierarchy, specifying an element in the **Element Search Root** field can improve search performance.

- **All Descendants**

  Select **True** to retrieve any sub-element in the hierarchy that matches the specified criteria. Select **False** to retrieve only root-level elements that match the specified criteria.

- **Template**

  Select the template that retrieved elements must be based on. After you select a template, you can add criteria to find elements by attribute value.

- **Category**

  Select the category that retrieved elements must match.

- **Attribute Value**

  You can specify up to three conditions for an attribute value. For each condition, specify an attribute name, operator, and attribute value, such as `Temperature >= 98`.

  - In versions prior to PI AF 2018, only available when you specify a template. For more details, see Configure search conditions for attribute values when a template is specified.
  - In PI AF 2018 and later versions, you can specify attribute values without needing to specify a template. For more details, see Configure search conditions for attribute values when no template is specified.

- **Description**

  Enter a string (of up to 440 characters) that retrieved elements must match.

- **Element Type**

  Select the type that retrieved elements must match.

- **Is Annotated**

  (PI AF 2017 or later versions.) Set to `True` to retrieve only elements that have annotations. Set to `False` to retrieve only elements that do not have annotations.

- **Creation Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve elements that were created in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Creation Date** a second time and filter the search for results between two values (`< *+1d` is the default).

- **Modify Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve elements that were modified in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Modify Date** a second time and filter the search for results between two values (`< *+1d` is the default).

> **Note:**
> An element's modify date is updated whenever an annotation or child element is added, as well as when a change to its configuration is checked into the database.
>
> Most template changes, or any modification to an attribute value that is not a configuration item, will not affect an element's modify date.

- **Results per Page**

  Enter the maximum number of elements to show on a single page of the search results.

6. Optional. Specify how you want results to be displayed in the **Results** table.

To ...	Do this ...
Group by template	Select the **Template** check box.
Group by category	Select the **Category** check box.
Change column selections	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Column Visibility**. c. Select or clear column selections as needed.
Display attributes as columns	a. Click and click **Select Attributes**. b. In the Select Attributes window, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) attributes. c. Click . d. Click **OK**.
Display full paths of elements	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Show Full Paths**.
Conceal full paths of elements	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Hide Full Paths**.

7. Select any of the results you want to use and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Topics in this section

- Configure search conditions for attribute values when a template is specified
- Configure search conditions for attribute values when no template is specified

Configure search conditions for attribute values when a template is specified

You can restrict your search based on the value of an attribute. After you specify a template (required in versions prior to PI AF 2018), use the **Attribute Value** field to configure up to three conditions that the search must match regarding an attribute value.

Before you start

- Unindexed attributes can take a significant amount of time to evaluate, particularly if they are configured with a data reference.
- You cannot search for attributes with `Object` or `Array` value types.

Procedure

1. Click ... (**Display attribute choices**) and select an attribute from the list of possible attribute categories:

Option	Description
attribute a	Indexed attributes, including configuration attributes.
Configuration Attributes	Configuration attributes that are not indexed.
Other Attributes	Non-configuration attributes that are not indexed.

2. Click >=, and select a mathematical operator from the list.
   - For attribute value types `Single` and `Double`, queries do not support the `In` operator.
   - For attribute value types `String`, `Boolean`, and `Int64`, queries do not support the following operators:
     - < (less than)
     - > (greater than)
     - <= (less than or equal to)
     - >= (greater than or equal to)
3. Enter a value in the units specified by the default UOM in the attribute template.

> **Note:**
> For indexed attributes that store `String` value types, the search only uses the first 40 characters of the entered value.

Configure search conditions for attribute values when no template is specified

When using a PI AF 2018 or later client and connected to a PI AF 2018 or later server, you can restrict your search based on the value of an attribute, without first having to specify a template. Use the **Attribute Value** field to configure up to three conditions that the search must match regarding an attribute value.

Before you start

- When no template is selected, all attributes are searched, including those used in templates.
- When no template is selected, wildcard characters should not be used in the **Attribute Value** field.
- Unindexed attributes can take a significant amount of time to evaluate, particularly if they are configured with a data reference.
- You cannot search for attributes with `Object` or `Array` value types.

Procedure

1. From the **Add Criteria** list, select **Attribute Value**.
2. In **Attribute Value**, enter the name of the attribute to retrieve.
3. Click >=, and select a mathematical operator from the list.
4. Enter a value. If the value type is `Enumeration Set` or `Guid`, you also need to append `as` *`Enumeration_Set`* or `as Guid` to the search query displayed in the **Enter element criteria** field. For example, to search for assets with a Health Status attribute value of Error, you would enter the following as the **Attribute Value** criteria:

   To complete the search query, you would append the name of the Health Status enumeration set to the string already displayed in the **Enter element criteria** field:

   ```
   as 'Health Status'
   ```

   The completed search query in the **Enter element criteria** field would look as follows:

   Note:
   You also need to append `As String` or `As Numeric` whenever the value type of an attribute does not match what the query search value appears to be. For example, if the value type of an attribute were `String` and the search value were `55` (which looks like a numeric), you would need to append `As String` to the query.

   Note:
   You can use uppercase and lowercase combinations when you append to search queries: `As String`, `as string`, and `as STRING` are all equivalents. You can also use either `'` or `"` to enclose enumeration sets and strings that contain space characters.

Special characters in name searches

When searching for objects by name, such as element names, event frame names, or attribute names (when associated with a template), you can use special characters:

Special character	Purpose
*	Substitute any number of unspecified characters.
?	Substitute a single unspecified character.
[*xyz*]	Specify a set of characters (*x*, *y*, or *z*) to match.
[!*xyz*]	Specify a set of characters (*x*, *y*, or *z*) to preclude a match.
\	Ignore the subsequent special character and interpret as its actual character.
[*first-last*]	Specify a range of characters (from *first* to *last*) to match. For example, `a[a-c]` matches `aa`, `ab`, or `ac`, but does not match `ad` or `abc`.

Search for attributes on elements

You can search for an attribute or group of attributes. You may want to locate a particular attribute, for example, to configure a formula data reference or to assign it as a value to another attribute. (To search for a specific attribute value, choose an attribute value as search criteria for an element search.)

Procedure

1. In the navigator pane, click **Elements**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Attribute Search**.
   - In the browser, right-click the **Element Searches** collection and select **New Attribute Search**.
3. Set the Attribute Search window to find the desired attributes in the PI AF database:
   a. Under **Where**, set the fields to restrict attributes retrieved:
      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.
      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.
      - **Attribute category**

        Select the category that retrieved attributes must match.
      - **Attribute value type**

        Select the type of value that the retrieved attributes must store.

- **Maximum results**

  Enter the maximum number of matching attributes to retrieve.

b. In the **Element Criteria** area, set the fields to restrict the elements searched for matching attributes:

- **Search Root**

  Enter the element that you want to use as the root or base level for the attribute search. Type the exact name or click **Browse** ... to open the Element Browser window, where you can view the element hierarchy and select an element. You cannot include wildcard characters in the entered name. If you do not specify an element, you set the main level of the element hierarchy as the root. Depending on your PI AF hierarchy, specifying an element in the **Search Root** field can improve search performance.

  Select the **Search Sub-Elements** check box to search the entered root and any sub-elements. Clear this check box to search only the entered root.

- **Name**

  Enter the name of the elements in which you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

- **Description**

  Enter a string (of up to 440 characters) to retrieve only those elements that match.

- **Category**

  Select the category of the elements in which you want to search for attributes.

- **Template**

  Select the template of the elements in which you want to search for attributes.

- **Type**

  Select the type of the elements in which you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table.

   Alternatively, use the element tree under **Search results** to browse for attributes under particular elements, and then select the attributes to add them to the **Search results** table.

   Remember that the attributes available by searching and the attributes available by browsing the element hierarchy depend on the configuration properties of the attributes.

5. Select items in the search results list and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for event frames

Use the Event Frame Search window to retrieve event frame data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Event Frame Search**.
   - In the browser, right-click the **Event Frame Searches** collection and select **New Search**.
3. Set the Event Frame Search window to find the desired event frame or event frames in the PI AF database.

   Note:
   If you specify values for multiple criteria, the search returns only those event frames that match *all* the specified criteria.

4. From the **Actions** ▾ list, select the type of filter to apply: **Contains**, **Exact Match**, **Starts With**, or **Ends With**.
5. Choose from the following actions:

To ...	Do this ...
Type one or more filter conditions directly into the **Enter Event Frame Criteria** field	◦ Use the : or := operator to select elements that match a name, description, template, primary element name, or category. ◦ Use special characters as needed. For more information, see Special characters in name searches. ◦ Enclose strings that contain spaces or special characters with the " character. ◦ Separate filter conditions with a whitespace, for example: `Category:"Processing Plants" Name:Cracking`
Filter event frames under **Criteria**	Enter values in the following fields as needed. Click **Add Criteria** to enter criteria for additional fields.

   - **Search**

     Select the method you want to use to specify when the desired event frames occurred. The window shows the appropriate fields for the selected method.

     For example, select **Active Between** to specify a start time and end time, and find event frames active any time during that period. Select **Starting After** to specify only a start time, and find event frames that start after that time.

   - **In Progress**

     If available, select this check box to further restrict matching event frames to those that have not yet finished.

- **Search start**

  A PI time expression that specifies the start of the time period used to search for event frames.

- **Search end**

  A PI time expression that specifies the end of the time period used to search for event frames.

  Optional. From the list next to this field, select a defined time to automatically fill in the **Search start** and **Search end** fields.

- **All Descendants**

  Select this check box to search all levels of the event frame hierarchy below the specified root for matching event frames.

- **Name**

  Enter the name of the event frame to retrieve, based on the filter type. You can enter special characters to match part of a name. See Special characters in name searches.

- **Element Name**

  Enter a PI AF element that must be the parent of any retrieved event frames. You can enter special characters to match part of a name. See Special characters in name searches.

- **Category**

  Select the category that retrieved event frames must match.

- **Results per Page**

  Enter the maximum number of event frames to show on a single page of the search results.

- **Template**

  Select the template that retrieved event frames must be based on. After you select a template, you can add criteria to find elements by attribute value.

- **Analysis Name**

  Enter the name of the analysis that retrieved event frames were generated from. You can use wildcards as needed.

- **Attribute Value**

  You can specify up to three conditions for an attribute value. For each condition, specify an attribute name, operator, and attribute value, such as `Temperature >= 98`.

  - In versions prior to PI AF 2018, only available when you specify a template. For more details, see Configure search conditions for attribute values when a template is specified.
  - In PI AF 2018 and later versions, you can specify attribute values without needing to specify a template. For more details, see Configure search conditions for attribute values when no template is specified.

- **Duration**

  Select an operator and enter a value, which can include a PI time abbreviation. For example, select **>=** and enter `1d` to retrieve events that last at least one day. You can select **Duration** a second time and filter the search for results between two values. For example, select **<=** and enter `2d` to retrieve events that lasted between one and two days.

- **Event Frame Search Root**

  Enter the event frame that you want to use as the root or base level for the event frame search. Type the exact name or click **Browse** ... to open the Event Frame Browser window, where you can view the event frame hierarchy and select an event frame. You cannot include wildcard characters in the entered name. If you do not specify an event frame, you set the main level of the event frame hierarchy as the root. Depending on the complexity of your PI AF hierarchy, specifying an event frame in the **Event Frame Search Root** field can improve search performance.

- **Can Be Acknowledged**

  (PI AF 2016 or later versions.) Set to `True` to retrieve event frames that can be acknowledged. The ability to acknowledge event frames is determined in the template on which the event frame is based.

- **Is Acknowledged**

  (PI AF 2016 or later versions.) Set to `True` to retrieve event frames that have been acknowledged. Set to `False` to retrieve only event frames that have not been acknowledged.

- **Is Annotated**

  (PI AF 2016 or later versions.) Set to `True` to retrieve only event frames that have annotations. Set to `False` to retrieve only event frames that do not have annotations.

- **Severity**

  (PI AF 2016 or later versions.) Select an operator and select a severity setting from the list. For example, select **>= Minor** to retrieve event frames that have at least a Minor severity setting. You can select **Severity** a second time and filter the search for results between two severity settings. For example, select **<= Critical** to retrieve event frames that have Minor, Major and Critical severity settings.

- **Creation Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve event frames that were created in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Creation Date** a second time and filter the search for results between two values (`< *+1d` is the default).

- **Modify Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve event frames that were modified in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Modify Date** a second time and filter the search for results between two values (`< *+1d` is the default).

> **Note:**
> An event frame's modify date is updated whenever a capture value, an annotation, or a child event frame is added, as well as when a change to its configuration is checked into the database.
>
> Most template changes, or any modification to an attribute value that is not a configuration item, will not affect an event frame's modify date.

6. Optional. Specify how you want results to be displayed in the **Results** table.

To ...	Do this ...
Group by template	Select the **Template** check box.
Group by category	Select the **Category** check box.
Change column selections	a. Right-click the column heading in the **Results** table or white space below. b. Click **Column Visibility**. c. Select or clear column selections as needed.
Display attributes as columns	a. Click and click **Select Attributes**. b. In the Select Attributes window, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) attributes. c. Click . d. Click **OK**.
Display full paths of event frames	a. Right-click the column heading in the **Results** table or white space below. b. Click **Show Full Paths**.
Conceal full paths of event frames	a. Right-click the column heading in the **Results** table or white space below. b. Click **Hide Full Paths**.

7. Click **Search** to retrieve the matching event frames into the **Results** table.
8. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for attributes on event frames

Use the Event Frame Attribute Search window to retrieve event frame attribute data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Event Frame Attribute Search**.
   - In the browser, right-click the **Event Frame Searches** collection and select **New Attribute Search**.

3. Set the Event Frame Attribute Search window to find the desired attributes in the PI AF database:

   a. Under **Where**, set the fields to restrict attributes retrieved:

      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.

      - **Attribute category**

        Select the category that retrieved attributes must match.

      - **Attribute value type**

        Select the type of value that the retrieved attributes must store.

      - **Maximum results**

        Enter the maximum number of matching attributes to retrieve.

   b. In the **Event Frame Criteria** area, set the fields to restrict the search to matching attributes:

      - **Search**

        Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

      - **Search start / Search end**

        Choose or enter start and end times for the search time period.

      - **Search Root**

        Enter the event frame to use as the root or base level for the attribute search. Type the exact name or click **Browse** ... to open the Event Frame Browser window, where you can view the hierarchy and select an event frame. You cannot include wildcard characters in the entered name. If you do not specify an event frame, the root is set to the main level of the hierarchy. Depending on the complexity of your PI AF hierarchy, specifying search root can improve search performance.

        Select the **Search Sub-Event Frames** check box to search the entered root and its children. Clear this check box to search only the entered root.

      - **Name**

        Enter the name of the event frames you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Description**

        Enter a description string of up to 440 characters to retrieve only those elements that match.

- **Duration**

  Enter minimum and maximum duration values to limit the attribute search to event frames whose duration is within these limits.

- **Category**

  Select the category of the event frames you want to search for attributes.

- **Template**

  Select the template of the event frames you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table. You can limit the results further to list only those attributes that match the text entered into the **Search results** field.
5. Select items in the search results list and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for transfers

Use the Transfer Search Criteria window to retrieve transfer data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Transfer Search**.
   - In the browser, right-click the **Transfer Searches** collection and select **New Search**.
3. Set the Transfer Search Criteria window to find the desired transfers in the PI AF database:
   - **Search**

     Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

   - **Search start / Search end**

     Choose or enter start and end times for the search time period.

   - **Name**

     Enter the name of the transfers you want to search for. You can enter special characters to match part of a name. See Special characters in name searches.

   - **Description**

     Enter a description string of up to 440 characters to retrieve only those transfers that match.

- **Source**

  Enter or click ... to open the Element Browser window, where you can view the element hierarchy and select the source element for transfers you want to include in the search. Click the **Any** button to reset to the default <Any>.

- **Destination**

  Enter or click ... to open the Element Browser window, where you can view the element hierarchy and select the destination element for transfers you want to include in the search. Click the **Any** button to reset to the default <Any>.

- **Template**

  Select the template of the transfers you want to search for.

- **Maximum results**

  Enter the maximum number of matching transfers to retrieve.

If you specify values for multiple settings, the search returns only those transfers that match all the specified settings.

4. Click **Search** to retrieve the matching transfers into the **Search results** table.

   You can limit the results further to list only those transfers that match the text entered into the **Search results** field.

5. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for attributes on transfers

Use the Transfer Attribute Search window to retrieve transfer attribute data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Transfer Attribute Search**.
   - In the browser, right-click the **Transfer Searches** collection and select **New Attribute Search**.
3. Set the Transfer Attribute Search window to find the desired attributes in the PI AF database:
   a. Under **Where**, set the fields to restrict attributes retrieved:
      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.

- **Attribute category**

  Select the category that retrieved attributes must match.

- **Attribute value type**

  Select the type of value that the retrieved attributes must store.

- **Maximum results**

  Enter the maximum number of matching attributes to retrieve.

b. In the **Transfer Criteria** area, set the fields to restrict the attribute search to transfers matching the specified criteria:

- **Search**

  Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

- **Search start / Search end**

  Choose or enter start and end times for the search time period.

- **Name**

  Enter the name of the transfers you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

- **Description**

  Enter a description string of up to 440 characters to retrieve only those attributes that match.

- **Source**

  Enter or browse to and select the source element for transfers whose attributes you want to include in the search.

- **Destination**

  Enter or browse to and select the destination element for transfers whose attributes you want to include in the search.

- **Template**

  Select the template of the transfers you want to search for attributes.

- **Category**

  Select the category of the transfers you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table.

   You can limit the results further to list only those attributes that match the text entered into the **Search results** field.

5. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Trends in PI System Explorer

While browsing elements in PI System Explorer, you can quickly create simple trends to verify the formulas, calculations, and other measurements you are building within PI AF. For example, while you are designing a PI AF formula, you might want to see what the results look like over the last several hours. Or, if you are creating PI point attributes that involve summaries, it could be useful to see what those look like over recent hours.

You can add the PI AF attributes or PI points of interest to a trend; one way of adding them is to drag and drop them onto the trend. You can then explore the data by adjusting the duration and the start and stop times of the trend.

After you have created the trend, it remains available in the Trend window even when you are using other features within PI System Explorer. You can add or remove traces to the trend at any point. You can also right-click the trend cursor and select commands from a context menu.

You can also see a trend of an attribute when you view its time series data (by right-clicking an attribute and clicking **Time Series Data**). The trend is automatically shown at the bottom of the Time Series Data window.

Create trends

To verify formulas, calculations, and other measurements in PI AF, create a trend.

Before you start

OSIsoft recommends you use visualization tools, such as PI Vision, PI ProcessBook, and PI WebParts, to create and save trends that you plan to use more than once.

Procedure

1. From the PI System Explorer menu, select **View** > **Show Trend**.
2. In the Trend window, choose from the following actions.

To ...	Do this ...
Add an attribute to the trend	a. Click **Add Attributes**. b. In the Attribute Search window, specify search parameters in the **Attribute Name**, **Attribute Category**, and other fields to find the attribute of interest. c. Expand the element hierarchy in the Search results area and select the attribute. d. Click **OK.** The Trend window displays a trend showing how that attribute has varied over the past hour (1 hour is the default duration).
Adjust the duration	Specify times in the **Start Time** and **End Time** fields.

Add a PI point to the trend	a. Click **Add PI Points**. b. In the Tag Search window, search for the PI point you want to see and select it in the results. c. Click **OK**. The PI point is added to the trend.
Adjust which traces are visible on the trend	Click **Traces** and clear the check box of attributes or points you do not want to see.

3. Add any further PI AF attributes and PI points for which you want to view data.

   The traces in the Trend window persist until you close the window.

   You can drag attributes and PI points from other search results and drop them on the trend to add them. Alternatively, you can right-click an attribute or point and click **Add to Trend**.

Version management for equipment and processes

When you need to make an equipment change, such as swapping out a broken pump for a similar pump, you can continue to use the existing element to represent the new pump. However, to indicate that a change has occurred, it is a good idea to create a new version of the element. That way you have a record of the change to the equipment that the element represents.

Similarly, you might need to create versions of a table. For example, strapping tables of tanks are updated periodically to match the changing physical characteristics of the tank. By versioning the table, you will be able to use the table as it was configured at the time of the data collection.

PI System Explorer allows you to create multiple versions of elements, models, and tables.

You can also indicate an element, model, or table is not in service past a specific date by selecting that date as its *obsolete* date.

Videos

For an overview of version management, watch this video:

https://www.youtube.com/watch?v=8cHFHW1SID0&rel=0

For information on how to create versions of elements and tables, watch this video:

https://www.youtube.com/watch?v=j2ilXV2XmV0&rel=0

Topics in this section

- Create versions for elements or models
- Create table versions
- Compare two versions

Create versions for elements or models

Create a new version when you need to keep track of changes to an asset.

Before you start

Whenever a new version is created, a check-in occurs on the element or model.

Procedure

1. In the navigator, click **Elements**.
2. in the Elements browser, choose from the following actions.
   - Right-click the element or model you want to version and click **Create Version**.
   - Click the element or model you want to version. In the viewer, select the **Version** tab and click **New Version**.
3. In the Create Version window, enter an **Effective Date** for the version or click to select a date. PI System Explorer creates a new version.
4. Optional. In the **Comment** field, enter information on the new version.
5. Click **OK**.

   To indicate that more than one version exists, the icon in the browser changes to for an element or for a model.

Create table versions

Create a new version when you need to keep track of changes to a table.

Before you start

Whenever a new version is created, a check-in occurs on the table.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, choose from the following actions.
   - Right-click the table you want to version and click **Create Version**.
   - Click the table you want to version. In the viewer, select the **Version** tab and click **New Version**.
3. In the Create Version window, enter an **Effective Date** for the version or click to select a date. PI System Explorer creates a new version.
4. Optional. In the **Comment** field, enter information on the new version.
5. Click **OK**.

   To indicate that more than one version of the table exists, the icon in the browser changes to .

Compare two versions

You can compare two versions of an element, model, or table that has more than one version.

Procedure

1. Choose from the following actions.

To...	Do this...
Compare table versions	a. In the navigator, click **Library**. b. In the Library browser, select a table that displays the icon.
Compare element or model versions	a. In the navigator, click **Elements**. b. In the Elements browser, select an element that displays the icon, or a model that displays the icon.

2. In the viewer, select the **Version** tab and click **Show History**.
3. In the Show History window, select two versions in the left panel from the list of all existing versions.

   The right panel uses colored change bars to indicate the differences between the versions:

   - Red means that the item was present in the older version and is not present in the newer version.
   - Green means the item is present in the newer version but was not present in the older version.
   - Yellow means that the item changed between the older and newer version.
4. To exit, click **Close**.

Display of different object versions and obsolete objects

By default, PI System Explorer shows the latest versions of elements, models, and tables (even if those versions are in the future). It also excludes earlier versions and all objects that have an **Obsolete** date specified (even if that date is in the future). As a result, when you specify an **Obsolete** date for an object and check it in, it immediately disappears from PI System Explorer.

You can, however, find and view an obsolete object, or a particular version of an object, by setting **Query Date** to a fixed time at which the object or version is current. To display an element that became effective January 1, 2010 and obsolete on January 1, 2011, for example, set **Query Date** to any time between those two dates.

Video

For information on how to display different versions, watch this video:

https://www.youtube.com/watch?v=MwFrhzjPrns&rel=0

Topics in this section

- Query Date and PI System Explorer time
- Show specific dates and times
- Time context for attribute values

Query Date and PI System Explorer time

Query Date determines the time values PI System Explorer uses to:

- Find and display attribute values.
- Identify current versions of objects to display and include in search results.

At the default **Query Date** setting, **Set to Latest**, PI System Explorer finds and displays the most current data for attribute values. You can override the **Query Date** to view values for a different time with **Set Time Context**. See Set time context for displayed attribute values.

Set to Latest also causes PI System Explorer to use the latest versions of objects, even if those versions are in the future, and excludes any object with an **Obsolete** date, even if that date is in the future.

You can change **Query Date** to specify a fixed date and time:

- The date and time picker lets you choose a specific date and time setting.
- The selections **Set to Now** and **Set to Today** each use the current time to establish a fixed date and time. For example, if you choose **Set to Now** on Monday, then on Wednesday **Query Date** will still be set to Monday.

How Query Date affects PI System Explorer

What is affected	Query Date is *Set to Latest*	Query Date is a fixed date and time
All PI System Explorer objects	PI System Explorer uses the latest versions, even if those versions are in the future; it excludes any object with an obsolete date, even if that date is in the future.	PI System Explorer uses versions of objects that are in effect at that time.
Element, model, and table versions in search	PI System Explorer returns the latest versions, even if those versions are in the future.	PI System Explorer returns the versions that are in effect at that time.
Relative times in searches	PI System Explorer defines the current time (*) as the time of the search.	PI System Explorer defines the current time (*) as the time specified by the Query Date.
PI data	PI data is returned at its latest snapshot value.	PI data returned is the value for the timestamp specified by the Query Date.

Show specific dates and times

When you set the query date to a specific date and time, it remains constant at that date and time until you change it again.

Procedure

1. On the PI System Explorer tool bar, click the **Query Date** button.
2. In the Date and Time Picker window, choose the date and time that you want. If you want to set the query date to the current date and time, click the **Set to Now** button.

> **Note:**
> When you choose **Set to Now** as the query date, you are setting the query date to a specific date and time; this is a constant value. This is not the same as setting the query date to always be the current time. For that, you should choose the **Set to Latest** option.

3. Click **OK**.

Video

For information on how to set a specific query date, watch this video:

https://www.youtube.com/watch?v=_381ZTzkFo4&rel=0

Time context for attribute values

The time or time range that PI System Explorer uses to display attribute values is determined by values you set in the Set Time Context window.

By default, PI System Explorer displays the latest values for attributes based on the current time. The Query Date reflects PI System Explorer's time setting (see Query Date and PI System Explorer time). You can view attribute values for a different time, however, by specifying a new time in the Set Time Context window.

The time and time range settings in the Set Time Context window apply only to displayed attribute values. Query Date continues to apply to all other PI System Explorer objects.

Set time context for displayed attribute values

You use Set Time Context to specify a different time or time range for displayed attribute values.

Procedure

1. On the PI System Explorer toolbar, click .

   Alternatively, click **Tools** > **Options** and select the **Time Context** tab.

2. In the Set Time Context window, select one of the following options. Enter time values for an option directly, or click to select a date and time. You can also click to construct a time expression, such as `*-1d`.

Option	Description
Query Date Time	Displays the current setting for Query Date (defaults to `Latest Available`). You can specify a different date time setting for Query Date, to enable PI System Explorer to evaluate which versions of objects to retrieve from the PI AF database.
Alternative Time	Displays the time context for data retrieval after applying the current Query Date setting. The default is the current time (*), but you can enter a new time context to use for displayed attribute values.

Time Range	Enter Start and End times to specify a new time range to use for displayed attributed values that require a time range.

3. Optional. Click **Set as Default** to register the current settings as the default time context, or click **Restore Default** to restore the time context for **Query Date Time** to `Latest Available`.
4. Click **Apply** to see your changes and **OK** to keep them. The revised time context settings are displayed in the PI System Explorer title bar.

View time series data

PI System Explorer can display time series data for attributes that contain or reference such data. This allows you to preview how your data reference configurations will work with your collected data in other PI client tools, such as PI ProcessBook, PI Vision, PI DataLink and PI WebParts.

Note:
Some attribute configurations do not support this type of data. See Restrictions on viewing time series data.

Procedure

1. Right-click on the attribute in the viewer and click **Time Series Data**.
2. In the Time Series Data window, choose the tab representing the type of data you want to see:
   - **Archive**

     For PI points. Shows the archived values within the specified time. Settings include Boundary type and Filter expression.
   - **Sampled**

     Returns evenly-spaced, interpolated values over a regular interval. You can include a Filter expression.
   - **Plot**

     Retrieves values over the specified time range suitable for plotting over the specified number of intervals. Typically, the intervals represent pixels and you would use this feature to represent the screen width available for the plot.
   - **Summary**

     Displays summary Statistics for attribute values that support this feature. You can specify Weighting.
   - **Data Pipe**

     Enables you to monitor and display data changes within attributes. For attributes that support future data, you can specify how the data pipe returns data with the Event horizon mode and Offset properties. For historical tags, Event Horizon Mode has no effect.

Topics in this section

- Boundary type

- Filter expression
- Statistics
- Weighting
- Event horizon mode
- Restrictions on viewing time series data

Boundary type

Specify a boundary type to determine how the searches for data values are handled near the start and end times of the value range:

- **Inside (default)**

  Returns values at start and end times, if they exist, or the nearest values occurring within the range.

- **Outside**

  Returns the closest values occurring immediately outside the range.

- **Interpolated**

  Returns interpolated values at start and end times.

Filter expression

Add a filter expression to filter event values using a mathematical expression, eliminating data for which the expression evaluates as false. For example, the simple filter expression:

```
'.' < 70
```

would remove all values over 70 from the calculation. You can also use any valid PI performance equation in the filter expression to build more complex expressions to remove atypical peaks in data values, for example.

Expression variables are limited to attributes or PI points which originate from a single PI Data Archive. Attributes which resolve to a static value (no data reference configured), are also acceptable. See Indirect PI point references for a complete description of possible reference syntax.

Attribute variable examples

Filter expression	What it does
`'.'`	Reference to the attribute being queried.
`'Level'`	Reference to the attribute `Level` at the same attribute hierarchy level in the element or event frame.
`'..'`	Reference to the parent attribute of the attribute being queried. Only valid for nested child attributes.
`'.|HighLimit'`	Reference to the child attribute `HighLimit` of the attribute being queried.
`'|Temperature'`	Reference to the `Temperature` attribute at the top hierarchy level in the current element or event frame.
`'\\MyPIServer\sinusoid'`	Absolute path to a PI point. It must be in the same PI Data Archiveas the current queried attribute.

Filter expression	What it does
`'\\myAFServer\myDatabase\myElement|myAttribute'`	Absolute path to a PI AF attribute.
`'\myRootElement|myAttribute'`	Database relative path to a PI AF attribute.

PI point variable examples

Filter expression	What it does
`'.'`	Reference to the PI point being queried.
`'sinusoid'`	Reference to the PI point `sinusoid` in the same PI Data Archive.

Statistics

The **Summary** tab provides the following statistics for attributes that support such statistics:

- **Percent Good**

  Displays the percentage of time for which good values are returned over the total time range. Good values are event values determined to be valid and not in an error state.

- **Average**

  Computes the average of values during the interval.

- **Minimum**

  Returns the minimum value during the interval.

- **Maximum**

  Returns the maximum value during the interval.

- **Range**

  Computes the maximum value minus the minimum value during the interval.

- **Standard deviation**

  Computes the standard deviation of values during the interval.

- **Population standard deviation**

  Computes the population standard deviation of values during the interval.

- **Count**

  Returns the number of values stored during the interval.

Weighting

The **Summary** tab allows you to select the Weighting for the statistical calculations:

- **Time-weighted**

  Default. Weights each event value by the length of time over which it applies.

- **Time-weighted continuous**

Time-weights values, but does all interpolation between values as if the values represent continuous data, (standard interpolation) regardless of whether the attribute is stepped.

- **Time-weighted discrete**

  Time-weights values, but does all interpolation between values as if the values represent discrete, unrelated values (stair step plot) regardless of the attribute is stepped.

- **Event-weighted**

  Weights each event equally. This method requires at least one event in a time range (two events for standard deviation calculations). By default, events at the boundary of the calculation are handled as follows:

  - use events at both boundaries when there is only one calculation interval
  - include events at start time in multiple intervals and the intervals are in ascending time order
  - include events at the end time in multiple intervals and the intervals are in descending time order

- **Event-weighted - Exclude Earliest Event**

  EventWeighted, except that the event at the start time (earliest time) of an interval is not used in that interval.

- **Event-weighted - Exclude Most Recent Event:**

  EventWeighted, except that the event at the end time (most recent time) of an interval is not used in that interval.

- **Event-weighted - Include Both Ends:**

  Events at both ends of the interval boundaries are included in the event-weighted calculation.

Event horizon mode

Event horizon mode controls when events for attributes that support future data are returned in the PI AF data pipe. Possible values are described in the following table.

Value	Description
End of Stream	Events are returned as they created, without regard to the current time.
Time Offset	Events are only returned after they cross the relative time offset into the future from now. For example, with a time offset of zero, future tag data changes are not returned until their timestamp is less than or equal to the current time.

Value	Description
Time Offset with Outside Event	Events are only returned if they occur before the relative time offset into the future, or they are the next event that would cross this time horizon. For example, with a time offset of one hour, future tag data changes are not returned until their time stamp is less than or equal to the current time plus one hour, or they are the next known event after the one hour time horizon. This mode is useful for trending data into the future.

Restrictions on viewing time series data

PI point data reference configuration types

Although many configurations are possible with PI point data references, support differs for each data function. PI point data references can be grouped into the following types for time method configurations:

- **Simply configured PI point**

  All data functions are supported and there is no additional configuration. Time method is Automatic, Interpolated, or AtOrBefore.

- **Archive retrieval of specified PI point**

  Only functions in the interpolated values group (listed in the following table) are supported. Time method is Before, Exact, AtOrAfter, or After.

- **Relative time PI point**

  Simply configured PI point with relative time configured. Time method is Automatic, Interpolated, or AtOrBefore.

- **Time range only**

  Time method is not supported.

- **Time range with relative time**

  Time method is TimeRange or TimeRangeOverride.

The following table details which data functions are supported by the PI point data reference configurations listed above. These restrictions also apply to other client tools, such as PI DataLink, PI Vision (formerly known as PI Coresight), PI ProcessBook and PI WebParts.

Data function	Simply configured	Archive retrieval	Relative time	Time range only	Time range with relative time
`InterpolatedValue`	✓	✓	✓	✕	✓
`InterpolatedValues`	✓	✓	✓	✕	✓
`InterpolatedValuesAtTimes`	✓	✓	✓	✕	✓

Data function	Simply configured	Archive retrieval	Relative time	Time range only	Time range with relative time
`RecordedValue`	✓	✕	✓	✕	✕
`RecordedValues`	✓	✕	✓	✕	✕
`RecordedValuesAtTimes`	✓	✕	✓	✕	✕
`RecordedValuesByCount`	✓	✕	✓	✕	✕
`Async`	✓	✕	✓	✕	✕
`PlotValues`	✓	✕	✓	✕	✕
`Summary`	✓	✕	✓	✕	✕
`Summaries`	✓	✕	✓	✕	✕
`FilteredSummaries`	✓	✕	✕	✕	✕
`UpdateValue`	✓	✕	✓	✕	✕
`UpdateValues`	✓	✕	✓	✕	✕
`Annotations`	✓	✓	✓	✕	✕
`Filters`	✓	✕	✕	✕	✕
`AFDataPipe`	✓	✕	✕	✕	✕

Non-PI point data reference configuration types

Non-PI point data references can be grouped into the following types for time method configurations:

- **No data reference**

  Data reference is set to None.

- **Calculation-based data reference with time-series inputs**

  Calculation-based data references with time-series inputs, such as Analysis, Formula, String Builder, Table Lookup, and URI Builder support the base data functions. However, filtering and annotations functions are not supported. Calculation-based data references use recorded values of their inputs to generate timestamps for their own recorded values. If calculation-based data references contain no time-series inputs, they behave like the No-data-reference configuration type, except that they cannot be written to.

  > **Note:**
  > Table Lookup data references that are configured with a time column, as described in Table provided time series data, support the same data functions as the configuration type: Custom data reference generating its own time-series data.

- **Custom calculation-based data reference**

  Custom calculation-based data references that implement their own time-series calls are required to "opt-in" to gain access to base data functions.

- **Custom data reference generating its own time-series data**

  Custom data references that generate their own time-series data require code to be implemented for each data function call.

The following table details which data functions are supported by the non-PI point data reference configurations listed above. These restrictions also apply to other client tools, such as PI DataLink, PI Vision, PI ProcessBook and PI WebParts.

Data function	No data reference	Calculation-based data reference with time-series inputs	Custom calculation-based data reference	Custom data reference with its own time-series data
`InterpolatedValue`	✓	✓	OptIn[1]	Code[2]
`InterpolatedValues`	✓	✓	OptIn[1]	Code[2]
`InterpolatedValuesAtTimes`	✓	✓	OptIn[1]	Code[2]
`RecordedValue`	✓	✓	OptIn[1]	Code[2]
`RecordedValues`	✓	✓	OptIn[1]	Code[2]
`RecordedValuesAtTimes`	✓	✓	OptIn[1]	Code[2]
`RecordedValuesByCount`	✓	✓	OptIn[1]	Code[2]
`Async`	✓	✓	OptIn[1]	Code[2]
`PlotValues`	✓	✓	OptIn[1]	Code[2]
`Summary`	✓	✓	OptIn[1]	Code[2]
`Summaries`	✓	✓	OptIn[1]	Code[2]
`FilteredSummaries`	✗	✗	✗	Code[2]
`UpdateValue`	✓	✗	✗	Code[2]
`UpdateValues`	✓	✗	✗	Code[2]
`Annotations`	✗	✗	✗	Code[2]
`Filters`	✗	✗	✗	Code[2]
`AFDataPipe`	✓	✓	OptIn[1]	Code[2]

[1] Indicates that a custom data reference can be written with one method to declare which attributes it uses as inputs, and another method that transforms inputs to outputs. With input support, the AF SDK uses these methods to emulate the others, using a time series generated from input data.

[2] Indicates that code is needed for each method, if the time series comes from the system of record that is being connected to.

Organization of asset models in PI AF

The goal of your asset model organization is to make assets easy to find for your users. The main method of organization is the element tree. PI AF elements are organized in a tree structure. Individual elements can be organized and regrouped within the tree, without limitations. If you are just getting started with PI AF, then start by creating the element hierarchy.

You can additionally organize through categories to speed up and simplify browsing and searching information. Think of categories as labels that you can apply to PI AF objects. Each object can have multiple categories.

Note:
You can also organize assets within a process. See Process models in PI AF.

Topics in this section

- The structure of PI AF asset models
- The design of PI AF asset hierarchies
- Element references in the asset hierarchy
- Categorization of objects

The structure of PI AF asset models

PI AF objects are organized in a tree structure, similar to the file structure on a Windows computer. In Windows, rather than having thousands of files on your desktop, you typically group files under folders. The same concept applies to PI AF elements. Organizing elements into hierarchies makes navigation of the elements easier, and it also provides insights into how the elements relate to one another.

When you create an asset model, you need to decide on a structure that makes it easy for users to find the different assets. Consider who your users are and what they will be looking for. For example, maintenance engineers might want to use PI System Explorer to find and record maintenance information. For this audience you might want to group assets by equipment type.

For example, if you had three pump elements, you might create an element called Pumps and then place all the pump elements beneath it. If you had two elements representing tanks, you might put them all under a Tanks element.

Asset model organized by equipment type

On the other hand, if you have multiple plants in different locations, that same maintenance engineer might want to see all the equipment located at his own plant. The following illustration shows the same elements organized by plant.

Asset model organized by location

You are not restricted to only one organizational strategy. You can use an *element reference* to include the same asset in more than one place in the tree. For example, you could choose to organize by equipment type and by plant as well. In the following illustration, the hierarchy includes the geographical tree and the equipment tree side by side.

Mixed asset model

You could alternatively nest the equipment organization under the geographical organization.

The design of PI AF asset hierarchies

The goal is to create an asset (element) hierarchy that is going to make sense to the people who need to use it. If you have different types of users, then you might need more than one tree structure.

If you are just getting started, do not try to do everything at once. Create a hierarchy for a subset of your assets. For example, you might start by modeling all your tanks, or alternatively, your equipment in a single plant, or your equipment from a particular manufacturer. Another approach would be to create a hierarchy for a particular type of user.

Gather information:

- What assets will be included in the tree? In other words, what types of equipment do you want to model?
- Who is going to need to find assets in this tree? Maintenance engineers? Process control engineers? Operators? For each type of user, what tasks will they need to perform?
- What assets are important to each user and what types of information will they need? Consider asking a few representative users of each type about what data they need to access and how. This should inform your organizational strategy.

Again, start small. You might start with one type of user. For example, suppose maintenance engineers need to use the model. If you have several plants, each with a group of maintenance engineers responsible for the equipment at that plant, then you should probably include a tree that organizes equipment by plant.

From there, you might ask some maintenance engineers how they would want to access the equipment information. Perhaps they usually look for assets by equipment type but sometimes they need to search by manufacturer. You could create parallel trees, one organized by equipment type and another by manufacturer. Each asset would then appear in each tree. Or you could organize the tree by equipment type and then use categories to label each asset by manufacturer.

After you create the hierarchy for a type of user, you might have a few of them try it out for a period of time, then take their feedback to improve the hierarchy. You can always change a hierarchy.

Note:
When thinking about users, remember that the element hierarchy might also be exposed in certain PI client applications. Consider the users of those applications as well. For example, PI Vision exposes the tree for users searching for related assets. In PI Vision, related assets are elements built from the same template.

Videos

For information on how to create elements, watch this video:

https://www.youtube.com/watch?v=d-kPQGabJuY&rel=0

For information on how to build element hierarchies, watch this video:

https://www.youtube.com/watch?v=hpiQSzubDu8&rel=0

Element references in the asset hierarchy

You can use element references to add the same element to your element hierarchy more than once, as described in The structure of PI AF asset models. When you create a reference to an existing element, you essentially create a placeholder in the hierarchy that points back to the referenced element. This allows users to find the element in all the relevant contexts, but it does not create a copy of the element. The exact behavior depends on the reference type.

Topics in this section

- Reference types
- Create single element references
- Create multiple element references

- Locate other references to the same element
- Change reference types
- Creation of custom reference types

Reference types

When you add an element reference to an element hierarchy the exact relationship between the referenced element and the element to which you add it depends on the type of reference. PI AF provides three system-defined reference types:

- Parent-Child
- Composition
- Weak Reference

You can also create your own custom reference types, as described in Creation of custom reference types and Child templates.

Videos

For information on how to use reference types, watch this video:

https://www.youtube.com/watch?v=OL43oUyuFEI&rel=0

For information on how to build custom reference types, watch this video:

https://www.youtube.com/watch?v=VcXsWfUEYQ8&rel=0

For the conclusion of the reference type example, watch this video:

https://www.youtube.com/watch?v=Vwta8GG4Jgk&rel=0

Topics in this section

- Parent-Child reference type
- Composition reference type
- Weak reference type

Parent-Child reference type

The default reference type is parent-child. The reference type is strong since it allows the child element to have many parents and exist as long as it has one or more strongly related parents. When the last parent element of the child is deleted, the child itself is deleted. Use the parent-child reference type when you want the child element to exist as long as it has a strong reference to at least one parent, but you do not want the child element to be treated as a single unit with the parent element.

Effect of a deleted parent-child reference when that parent-child reference type exists elsewhere

Example

A meter belongs to a company and is attached to a building. You would use a parent-child reference between the `Company` element and the child `Meter` element, and another parent-child reference between the `Building` element and the child `Meter` element. If the reference between the `Meter` and the `Building` element is deleted, the `Meter` element continues to exist because it has a parent-child reference to the `Company` element. However, if the child `Meter` element is also deleted from the `Company` element, it no longer exists because all parent-child references have been removed.

Composition reference type

A reference type with a reference strength of composition means that the child element is really a part of the parent and does not exist without the parent. If you delete a parent element that has a child element that is compositionally referenced, you delete the child element also.

Effect of a deleted parent element on elements with a composition reference type

Example

Use a composition reference when the two objects in the relationship are considered one item. For example, a meter might be composed of two sensors, and so you would use a composition reference between the `Meter` element and each of its two child `Sensor` elements. When you delete the `Meter` element, the child `Sensor` elements are also automatically deleted.

Weak reference type

Use a weak reference between two elements when you want to create a relationship between two elements but you do not want that relationship to control the lifetime of the child element. For example, you may want to organize your meters into groups, but if all strong references to the meter are removed, then you want it to automatically be removed from the grouping. In this case, you would use a weak reference between the group parent element and the child meter element.

Effect of a deleted strong element on an element with a weak reference type

Example

A child element with a weak reference is deleted as soon as no more strong references exist. You may find this useful if you want one view of your assets to be the master view. For example, in the above illustration the master view might be the `Site ABC` view and contain elements with strong parent-child reference types. Other views, such as `Pumps`, reference elements with a weak reference type. If the strong `Pump01` element in the master `Site ABC` view is deleted, the weak-referenced `Pump01` element in the `Pumps` view is deleted automatically.

Create single element references

You create a single element reference to establish a hierarchical relationship between two elements.

Procedure

1. In the Elements browser, click the element that you want to reference and drag it to the appropriate parent element.
2. In the Choose Reference Type window, select the reference type.

   If you are not sure what reference type you want, you probably want the default reference type, parent-child. See Reference types for more information.
3. Click **OK**.

   The referenced elements are displayed in the browser. The referenced element icon indicates that they are references.

Video

For additional information on how to create an element reference type, watch this video:

https://www.youtube.com/watch?v=Q1kK71JrhLo&rel=0

Create multiple element references

You create multiple element references to establish a relationship between an element and other elements in the hierarchy.

Procedure

1. In the Elements browser, right-click an element and click **New** > **Add Element Reference**.

   Note that you can also select an element and drag it to the parent.
2. In the Add Element(s) window, click the **Find Multiple Elements** button.

3. In the Element Search window, select search criteria to locate the desired elements and click the **Search** button.

   The search results field shows the results of your search.

4. Select the elements that you want to reference and click **OK**.
5. In the Add Elements window, select the reference type. If you are not sure what reference type you want, you probably want the default reference type, parent-child. See Reference types for more information.
6. Click **OK**.

   The referenced elements are displayed in the browser. The referenced element icon indicates that they are references.

Locate other references to the same element

A single element can be referenced in multiple places in a hierarchy. You can find all the elements in the hierarchy that contain the element or a reference to the element.

Procedure

1. Select the element in the Elements browser.
2. In the **General** tab, click the **Parents** link.

   The element's parents are displayed in the Parents of *Element* window.

3. Click **Close**.

Change reference types

You can change the reference type of a child element and event frame that has at least one parent in the hierarchy and is not at the root level of the hierarchy (where the reference type must be parent-child).

Procedure

1. Choose from the following actions:

To ...	Do this ...
Change a child element reference type	a. In the navigator, select **Elements**. b. In the browser, expand the Elements tree and locate the child element with more than one parent. c. Right-click the child element and click **Convert** > **Change Reference Type**.
Change a child event frame reference type	a. In the navigator, select **Event Frames**. b. In the browser, expand the Event Frames tree and locate the child event frame with more than one parent. c. Right-click the child event frame and click **Convert** > **Change Reference Type**.

2. In the Change Reference Type window, select a parent from the **Parent** list.

3. Select a new reference type from the **New Reference Type** list. Only valid reference types for the selected parent are available.
4. To establish the selected parent as the primary parent (when the selected reference type is valid), select the **Primary Parent** check box.

   Note:
   To be valid, the selected reference type must be strong and no composition references can be defined for the element or event frame.

5. Click **OK**.

Creation of custom reference types

Custom reference types are easy to create. Each custom reference type must be based on one of the three pre-defined reference types. Through the use of custom reference types, you can create rules that allow PI AF to guide users when they create new elements. See Child templates for information on creating template references.

Create custom reference types

Custom reference types are created in child templates. You can also create them to define your own hierarchy of elements using a more specific relation. Custom reference types must be based on one of the pre-defined reference types.

Note:
You can create reference types more quickly in the Element Templates collection using referenced templates, as described in Create child template references.

Procedure

1. In the Library browser, right-click the **Reference Types** collection and select **New Reference Type**.
2. In the **Name** field, enter a name for the reference type.
3. In the **Child Name** and **Parent Name** fields, enter names for the child and parent.
4. Optional. In the **Categories** field, click and select one or more categories to which the reference type belongs from the Categorize window. You can also enter one or more reference type categories directly, separated by a semicolon.
5. In the **Reference Strength** field, select **Strong**, **Weak**, or **Composition** from the list. See Reference types for definitions of reference type strengths.
6. In the **Allowed Parent Element Template** and **Allowed Child Element Template** fields, select a template to which the reference type parent and child belong from the list.

Categorization of objects

PI System Explorer allows you to organize objects into categories. Categories are essentially object groups that you define yourself. Their purpose is to help you find objects more easily. When you search for an object, you can use the category as a filter to reduce the list of results. Define as many categories as you like. Objects can belong to multiple categories.

For example, suppose you have a set of elements representing tanks. Half of the tanks are manufactured by ACME company, and the other half are manufactured by EMCA company. To locate tanks by manufacturer, create an ACME category and an EMCA category.

Each object type has its own categories. You cannot apply categories from one object type to an object of another type. For example, you cannot apply an element category to a table. PI AF supports the following category types:

- Analysis
- Attribute
- Element
- Notification Rule
- Reference Type
- Table

Videos

For information on how to create categories, watch this video:

https://www.youtube.com/watch?v=ux3qiQVSePU&rel=0

For information on how to assign categories, watch this video:

https://www.youtube.com/watch?v=5wkvG-1jOgM&rel=0

For information on how to group by category, watch this video:

https://www.youtube.com/watch?v=A0OlJZ19qWA&rel=0

Topics in this section

- Create new categories
- Assign objects to categories

Create new categories

Create categories for specific object types. Alternatively, use PI Builder to create multiple categories in an Excel worksheet.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, under **Categories**, right-click on the category object type and select **New Category**.
3. In the *Object_Type* Category Properties window, enter a name for the category in the **Name** field.
4. Optional. Enter a description of the category in the **Description** field.
5. Click **OK**.
6. Optional. Click the **Security** link if you wish to set up custom access permissions to the category beyond those already established at the server, database, or collection level. For more information, see Configure security for objects.

Assign objects to categories

Assign objects to the categories that you have created for each category type.

Procedure

1. In the viewer, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) objects that you want to categorize.
2. Right-click and select **Categorize**.
3. In the Categorize window, select the category or categories to assign to the selected objects.
   a. If the category you need is not listed, click the **New Category** button.
   b. In the Attribute Categories Properties window, create the new category and click **OK**.
4. Click **OK**. The categories are applied and are displayed in the **Categories** field in the configuration panel for each object.
   a. If any selected objects are created from templates, you can modify the template or not add the category to that particular object.

   Choose one of the following options:

   - Click **Modify the template and add the category** to modify all attributes derived from the template with the category.
   - Click **Do not add the category** to prevent the category from being assigned to the object.

   b. To apply the option to other selected attributes that are derived from templates, select the **Apply choice to remaining templates** check box.
   c. Click **OK**.

Representation of assets in PI AF

Each physical object that you include in your model is represented by a PI AF element. Physical objects are typically pieces of equipment, such as pumps or tanks. To associate data with an element, you create attributes on the element. Attributes can hold simple values representing fixed information, such as the diameter of a tank. An attribute can alternatively reference a PI point, a formula, a value from an external relational database, and more.

Templates

You can create each element individually, but OSIsoft recommends that you base individual elements on an asset template that represents the type of equipment. Element templates enable you to:

- Configure the template once; no need to individually configure each element based on the template.
- Update element structure across all elements belonging to the template.
- Maintain consistency in the asset model.
- Enable powerful features in PI AF client applications.

Topics in this section

- Asset representation with templates
- Template strategy
- Element templates
- Child templates
- Asset representation with elements
- Storage of application-specific information

Asset representation with templates

PI AF allows you to base similar objects on a single template. Templates essentially define a set of base attributes for all the objects that use that template. Create the template once and you can create as many elements based on the template as needed. If you make a change to a template, the change is automatically reflected in all the elements that use that template.

For example, suppose you have 100 pumps with the same three attributes. You can create a template for one pump and then base all the other pumps on that one template. The attributes in the template are automatically created for you in the pumps that are based on that template.

Now, suppose you need to make a change to the pump objects. You simply make the change in the template, and PI AF automatically propagates the change to all the pump objects that are

based on that template. Templates are a powerful tool, not only for creating new objects, but for keeping existing objects consistent and up to date.

A further advantage is that visualization tools can provide special features for objects based on templates. For example, suppose you build a trend for a pump based on a template. A visualization tool might let you swap or add in any other pump that is based on the same template. Assets based on the same template are sometimes called *related assets*.

Note:
You can also create templates for event frames, transfers, models, and notifications.

Template strategy

In almost every case, it is best to base your elements on templates. You not only save time but also ensure that you have consistent definition across all of the elements based on that template. Any changes that you make to an element template are propagated to every element that is based on that template. A single template modification can alter hundreds of elements. This allows you to make changes to your model in a single place; you don't need to update every element.

You do not have to plan and create all your templates at once. A good approach is to start by modeling a single type of asset. Create a template for the asset type. Decide what data, calculations, and other properties you need to store for that type of asset. Each of these items requires an attribute template.

Base templates

Template usage can be very broad or very specific. A template can define a specific type of measurement device, such as a brand-name instrument, or it can be a broad-use template specifying a particular role, such as a liquid mass meter. Depending on how broadly you define the template, you might find that the list of attributes is slightly different for different subsets of assets. In this situation, consider using a base template (for more information, see Base and derived templates). The alternative is to use a different template for each asset subset.

Extensions

You can also set up element templates to include attributes (as well as ports) that are not defined or maintained by the template (for more information, see Extensions). You can add simple, asset-specific attributes without the need to add them on all instances of the template.

Note:
Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.

Element templates

An element template is essentially a model of a type of asset. Element templates make creating displays, notifications, calculations, and analytics much simpler because equipment of the same type can share the implementation. Typically you create an element template that represents a type of equipment, such as a tank or pump. You can easily create elements based on the template because most of the element configuration is defined in the template.

When you change an element template, those changes propagate to all elements derived from it.

Deleting an element template deletes any notification or analysis templates that target it.

Note that:

- An element derived from a template gets its definition of categories, attributes, and ports from the template.
- If the template allows extensions, then a derived element can contain extra categories, ports, and attributes that are not defined in the template.
- Element templates can specify the allowed parent and child references (Parent Reference Types and Child Reference Types) for an element created from the template. This restricts the allowed relationships between elements in the hierarchy.

Video

For information on how to create element templates, watch this video:

https://www.youtube.com/watch?v=VYWXctstVg0&rel=0

Topics in this section

- Default attribute
- Base and derived templates
- Extensions
- Naming patterns
- Create element templates
- Find template references
- Define templates for other objects

Default attribute

The default attribute is an attribute that is most representative of the element. For example, if a user were to reference a tank, perhaps `Level` would be the single best attribute to use. For an electrical meter, the best attribute to use could be `Voltage`. The attribute must be at the top level of the attribute hierarchy of the element.

Notifications and asset analytics can take advantage of the default attribute to simplify some tasks. For example, if you were to create a comparison between two electrical meters, and had not specified which attributes to compare, the comparison would be performed using the `Voltage` attribute of the meters because you had made `Voltage` the default attribute.

Base and derived templates

A base template is best used when you are modeling elements that share most attributes, but have a few attributes that differ. A base template passes on all its attributes (as well as ports and analyses) to any derived templates that are created from it. When you later create an element from such templates, that element contains not only every attribute (as well as ports and analyses) from the base template, but also those from the derived template.

Base template example

Suppose you have a set of tanks, some with two valves and some with one valve: you can create an element template for the one-valve models, and use that as the base template for the derived template that you create for the two-valve models. When you create objects based on the two-valve model, they inherit all the attributes from the one-valve model, as well as any additional attributes defined for the two-valve model.

Derived templates

You define derived templates from a base template. You can set the base template by selecting from a **Base Template** list in the **General** tab of the element template you are defining, or you can right-click on an element template that is already displayed in the Library browser and click **New** > **New Derived Template**. The new template is immediately added to the Library browser with a new name ***Base_Template*Derived1**. You can then refine the definition, including the name, as described in Create element templates.

Override of base-template attribute values

Situations may exist where you need to use attribute template values in a derived template that suppress or override attribute template values inherited from their base template. Consider the following situations:

- Different substitution patterns are needed for PI points on the same type of asset (such as trucks with a common set of measurements, but truck manufacturer A uses a different PI point naming pattern than manufacturer B).
- Some assets employ direct measurement for an attribute PI point, whereas others may need that attribute value to be calculated. Similarly, an attribute value may be calculated for both types of assets, but the calculation is slightly different between the two.
- Different default UOMs or categories are needed for the same attribute, based on template.
- Not all attribute templates in a derived template are applicable (for example, some meters may only need three of the five attributes in a generic base `Meter` template).

You override base template attribute values by creating an attribute template with exactly the same name in the derived template. You can then modify properties as needed.

You can also prevent an attribute template that has been inherited from a base template from being visible to users or a client application, such as PI Vision, by creating an attribute template of the same name in the derived template and selecting **Excluded** as its **Properties** setting. For more information, see Excluded attribute property.

Template inheritance

By default, element templates are arranged by name in the Library browser. To determine the relationship that exists between templates, you might prefer to arrange them by template inheritance instead. You can right-click the **Element Templates** collection and click **Arrange By** > **Arrange By Template Inheritance**. Expand templates with ⊞ beside them to reveal derived templates.

Extensions

You can allow some attributes to be defined in the element itself, independent of the template. To do this you configure the template to allow extensions. Attributes that are created as extensions do not change when the template changes.

> **Note:**
> If you have a template that allows extensions and you later change it to disallow extensions, no new extended element attributes can be added to elements based on that template. However, all existing extended element attributes remain.

Extensions permit flexibility in cases where assets are similar in many areas, but a number of small variations exist. For example, suppose you build a template for a specific model of a car. All the cars of this model have the same core set of features: four tires, a steering wheel, and so on. However, one car may have a spoiler while the other car may not. One might have air conditioning, while another does not. Extensions are intended to handle this type of variation.

When extensions are enabled, you can base all the elements on a template, while adding additional information to each element as required. However, if you have multiple elements that are very similar to the template definition, but all require the same extra attributes, a base template might be a better approach than allowing extensions.

> **Note:**
> Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.

Naming patterns

When you define naming patterns for element, event frame, transfer, and model templates, you can use the substitution parameters described in the following table. These naming pattern shortcuts are available in addition to the substitution parameters that are listed in the **Naming Pattern** field when you click ▸.

Naming pattern substitution parameters

Parameter name	Substitution
Description parameters	
%DESCRIPTION:*path***%**	The object's description represented by the path to the object. If the path is not specified, the description of the current object is used.
%\ELEMENTDESCRIPTION%	The description of the object's root element, or the description of the primary referenced element of the object's root event frame.
%..\ELEMENTDESCRIPTION%	The description of the object's parent element, or the description of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENTDESCRIPTION%**, to retrieve further ancestors.
%\EVENTFRAMEDESCRIPTION%	The description of the object's root event frame.
%..\EVENTFRAMEDESCRIPTION%	The description of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAMEDESCRIPTION%**, to retrieve further ancestors.
ID parameters	
%ANALYSISID%	The ID of the object's analysis.
%ATTRIBUTEID%	The ID of the object's attribute or attribute template.
%|ATTRIBUTEID%	The ID of the object's root attribute or attribute template.
%..|ATTRIBUTEID%	The ID of the object's parent attribute or attribute template. Use **..|** notation, such as **%..|..|ATTRIBUTE%**, to retrieve further ancestors.

Parameter name	Substitution
%DATABASEID%	The ID of the object's PI AF database.
%ELEMENTID%	The ID of the object's element, or the ID of the primary referenced element of the object's event frame.
%\ELEMENTID%	The ID of the object's root element, or the ID of the primary referenced element of the object's root event frame. Use **..** notation, such as **%..\..\ELEMENTID%**, to retrieve further ancestors.
%..\ELEMENTID%	The ID of the object's parent element, or the ID of the primary referenced element of the object's parent event frame.
%EVENTFRAMEID%	The ID of the object's event frame.
%\EVENTFRAMEID%	The ID of the object's root event frame.
%..\EVENTFRAMEID%	The ID of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAMEID%**, to retrieve further ancestors.
%ID:*path***%**	The object's ID represented by the path to the object. If the path is not specified, the ID of the current object is used.
%MODELID%	The ID of the object's PI AF model.
%SYSTEMID%	The ID of the object's PI AF server.
%TRANSFERID%	The ID of the object's PI AF transfer.
Name parameters	
%ANALYSIS%	The name of the object's analysis.
%DATABASE%	The name of the object's PI AF database.
%\ELEMENT%	The name of the object's root element, or the name of the primary referenced element of the object's root event frame.
%..\ELEMENT%	The name of the object's parent element, or the name of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENT%**, to retrieve further ancestors.
%\EVENTFRAME%	The name of the object's root event frame.
%..\EVENTFRAME%	The name of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAME%**, to retrieve further ancestors.
%NAME:*path***%**	The object's name represented by the path to the object. If the path is not specified, null is returned since it references the name of the current object.
%SYSTEM%	The name of the object's PI AF server.
%\TEMPLATE%	The name of the root template of the object's element or event frame.
%..\TEMPLATE%	The name of the parent template of the object's element or event frame. Use **..** notation, such as **%..\..\TEMPLATE%**, to retrieve further ancestors.
Path parameters	
%ELEMENTPATH%	The path of the base element, the element of an attribute, or the primary referenced element of an event frame.
%..\ELEMENTPATH%	The path of the object's parent element, or the path of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENTPATH%**, to retrieve further ancestors.

Parameter name	Substitution
%EVENTFRAMEPATH%	The path of the event frame or the element of an attribute of an event frame.
%..\EVENTFRAMEPATH%	The path of the object's parent event frame. Use `..\` notation, such as **%..\..\EVENTFRAMEPATH%**, to retrieve further ancestors.
Time parameters **Note:** You can specify a format string with all time substitutions. The format string is separated from the substitution name by a colon. For example, `%TIME:d%` uses the short date pattern, whereas `%UTCTIME:yyyy/MM/dd HH:mm:ss.fff%` specifies the format in full.	
%ENDTIME:*yyyy-MM-dd HH:mm:ss.fff***%**	The current local end time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters.
%UTCENDTIME:*yyyy-MM-dd HH:mm:ss.fff***%**	The current Coordinated Universal Time (UTC) end time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters.

Create element templates

Create and configure an element template to ensure consistent definitions for elements in your asset structure.

Procedure

1. In the Library browser, right-click the **Element Templates** collection and click **New Template**.
2. Adjust settings on tabs to configure the element template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing template. Select the base template from the list. For more information on base templates, see Base and derived templates.
Type	You can base the template on an existing template. Select the base template from the list. For more information on base templates, see Base and derived templates.
Categories	Optional. You can organize objects by grouping them into categories. To browse available categories or to create a new category, click . See Categorization of objects.
Default Attribute	Optional. Select a default attribute from the drop-down list. See Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%TIME:yyyy-MM-dd HH:mm:ss.fff%** The current local time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute, represented by the path. You can also use other naming pattern substitution parameters. For example, if you want an element path included in the naming pattern, enter `%ELEMENTPATH%` as a substitution parameter. For a complete list of naming pattern substitution parameters, see Naming patterns. Each element derived from the template will have a unique, identifiable name. To ensure that new elements created from the template have an incremental number, enter * at the end of any pattern you enter here. If left blank, PI System Explorer uses the element template **Name** field and adds an asterisk to add an incremental number as needed. Some substitution parameters may not be applied when a derived element is created. To ensure a derived element's name fully reflects the naming pattern, right-click the element and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters.
Allow Extensions	Select this checkbox to enable additional attributes (as well as ports) to be defined for an element, beyond those defined in its template. See Extensions. **Note:** Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Annotations	Click this link to add one or more annotations to an element. For more information, see Element annotations.
Location	Click this link if you wish to set up location attribute traits for the element. However, you can also set them on an attribute if you prefer. Note that you can only assign one set of location attributes per element. For more information, see Attribute traits.

Option	Description
Security	Click this link if you wish to set up custom access permissions to the element template beyond those already established at the server and database level. For more information, see Configure security for objects.

3. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.
4. If you are creating a model, use the **Ports** tab to specify ports, which define end-points for connections between elements within a model. See Process models in PI AF.
5. Optional. Use the **Analysis Templates** tab to create an analysis template for the element template. For more information, choose from the following:
   - To create an expression analysis template, see Create an expression analysis template.
   - To create a rollup analysis template, see Create a rollup analysis template.
   - To create an event frame generation analysis template, see Create an event frame generation analysis template.
   - To create an SQC analysis template, see Create an SQC analysis template.
6. Optional. Use the **Notification Rule Templates** tab to select or create a notification rule template for the element template. For more information, see Create a notification rule template.
7. To save your work, press CTRL+S or click **Check In**.

Find template references

Locate where a template is used in PI System Explorer.

Procedure

1. In the Library browser, choose from the following options:
   - To locate an element template, expand **Templates** > **Element Templates** and select a template.
   - To locate an event frame template, expand **Templates** > **Event Frame Templates** and select a template.
2. Choose from the following options:

To find ...	Do this ...
Derived templates	Click **Derived Templates**. The Find Derived Templates for *Element_Template* window displays all element templates that have the selected element template as their base template. The list also includes element templates that are indirectly derived from the selected template, through multiple templates.

Elements created from the template	Click **Elements**. The Find Elements for *Element_Template* window displays all elements, models, transfers, and event frames that have been created directly from the selected element template. **Note:** To display the full path to elements, right-click on the column header (or below the search results grid) and click **Show Full Paths**.
Elements derived from the template	Click **Derived Elements**. The Find Derived Elements for *Element_Template* window displays all elements, models, transfers, and event frames that have been created from the selected element template, as well as from any template that is derived from it. **Note:** To display the full path to derived elements, right-click on the column header (or below the search results grid) and click **Show Full Paths**.
Referenced parent templates	Click **Referenced Parent Templates**. The Find Referenced Parent Templates for *Element_Template* window displays element templates that have specifically been linked to this element template as a parent through a reference type. The list of templates includes those that indirectly have been linked through template inheritance.
Referenced child templates	Click **Referenced Child Templates**. The Find Referenced Child Templates for *Element_Template* window displays element templates that have specifically been linked to this element template as a child through a reference type. The list of templates includes those that indirectly have been linked through template inheritance.
Event frames created from the template	Click **Event Frames**. The Find Event Frames for *Event_Frame_Template* window displays all event frames that have been created directly from the selected event frame template. **Note:** To display the full path to event frames, right-click on the column header (or below the search results grid) and click **Show Full Paths**.

Event frames derived from the template	Click **Derived Event Frames**. The Find Derived Event Frames for *Event_Frame_Template* window displays all event frames that have been created from the selected template, as well as from any template that is derived from it. **Note:** To display the full path to derived event frames, right-click on the column header (or below the search results grid) and click **Show Full Paths**.

Define templates for other objects

In addition to element templates, define templates as needed for event frames, models, and transfers. These templates have all the necessary attributes and properties for the role of the object. You then create instances of the object by basing them on the configured template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, open **Templates** to display the following object collections:
   - **Element Templates**
   - **Event Frame Templates**
   - **Model Templates**
   - **Notification Templates**
   - **Transfer Templates**
3. Right-click an object collection and select **New Template**.

   The new template displays the following tabs:
   - **General**
   - **Attribute Templates**
   - **Ports** (not available for event frame templates)
   - **Analysis Templates** (element templates only)
4. Fill in these tabs just as you would for an element template. See Create element templates.
5. To save your work, press CTRL-S or click **Check In**.

Child templates

In some cases, you might want to create a template that has one or more child templates. For example, suppose you have a template representing a motor and you want a separate template for the shaft. You want the shaft template to be a child of the motor template. You cannot directly create a child template in the same way that you would create a child element. Instead, you create a new referenced template below the `Motor` template.

This also creates a new reference type in the Reference Types collection.

Now, suppose you create an element based on the `Motor` template. PI System Explorer does not automatically create the child element, `Shaft`. However, when you manually create a child element on the motor element, the `Motor-Shaft` reference type is added to the list of reference types and the `Shaft` template appears in the Element Template list.

Create child template references

Create a child template reference when you need to establish parent-child relationships between two element templates.

Procedure

1. In the Library browser, right-click the template to which you want to add the child template reference and click **New** > **New Referenced Template**.
2. In the **Set Referenced Element Template Name** window, type a name for the child template, and click **OK**. This creates a child template in the **Templates** collection and a new reference type in the **Reference Types** collection.
3. Optional. To define a reference type other than strong parent-child, select the **Edit reference type** check box.
   a. In the Reference Type Properties window, select the desired reference strength from the **Reference Strength** list.
   b. Click **Check In**.
   c. Click **OK** to close the window.
4. Complete the element template definition, as described in Create element templates.

Asset representation with elements

In your asset model, elements represent physical objects, such as pumps or tanks. Each element has associated attributes that store properties and data for that element. Typically, you base each element on a template. For example, you would base each pump element on a pump template.

Topics in this section

- Create elements based on a template
- Create elements not based on a template

- Find where an element is used
- Deletion of elements
- Attribute-value reset to original properties
- Convert elements to element templates
- Change element templates

Create elements based on a template

Use the element templates that you have defined to create elements in your asset hierarchy.

Procedure

1. In the Elements browser, right-click the collection of elements or an individual element and click **New Element**.
2. In the **Choose Element Template** window, select an element template in the **Element Template** field.
3. In the **Add child element using the reference type** field, select a reference type from the list of available types.

   The available reference types depend on the selected element template and what reference types are defined in the database. If you are not sure what reference type to use, select the default reference type, parent-child.
4. Click **OK**.

   Property tabs for the new element are displayed in the viewer. Because the element is based on a template, most fields on the **General** tab are read-only (with gray or hash-pattern shading).
5. In the **Name** and **Description** fields, type a name and description for the element.
6. Optional. Perform the following actions as needed.
   - Click the **Extended Properties** link to create properties for client applications such as PI ProcessBook and PI WebParts. For more information, see Storage of application-specific information.
   - Click the **Location** link to set up location attribute traits for the element. For more information, see Set location attribute traits.
   - Click the **Annotations** link to enter a note for the element. For more information, see Element annotations.
   - Click the **Security** link to set up custom access permissions for the element. For more information, see PI AF object security.
7. Optional. Click the **Attributes** tab to review the attributes defined by the template. You cannot add new attributes unless the template allows extensions.
8. Optional. Click the **Notification Rules** tab to review any notification rules defined by the template. You cannot add new notification rules unless the template allows extensions.
9. To save your work, press CTRL-S or click **Check In**.

Create elements not based on a template

Although you can create elements that are not based on a template, OSIsoft recommends that eventually you base your elements on templates so that you can take advantage of powerful template-based features available in client applications. You can use the Convert to Template feature, as described in Convert elements to element templates.

Procedure

1. In the Elements browser, right-click the collection of elements or an individual element and click **New Element**.
2. In the **Choose Element Template** window, select a reference type in the **Add child element using the reference type** field.

   The available reference types depend on the element template and what reference types are defined in the database. If you are not sure what reference type to use, you probably want the default reference type, parent-child.
3. In the **Element Template** field, click **<None>**.
4. Click **OK**.

   Property tabs for the new element are displayed in the viewer.
   - Use the **General** tab to perform the basic element configuration.
   - The **Child Elements** tab is mainly used to view existing child elements on an existing element. You can optionally create new child elements directly from this tab.
   - Use the **Attributes** tab to create an attribute for each property or data item.
   - Use the **Ports** tab only if you are modeling a process.
   - Use the **Analyses** tab to select an analysis for the element.
   - Use the **Notification Rules** tab to select a notification rule for the element.
   - Use the **Version** tab to add version information for the element.
5. Optional. Perform the following actions as needed.
   - Click the **Extended Properties** link to create properties for client applications such as PI ProcessBook and PI WebParts. For more information, see Storage of application-specific information.
   - Click the **Annotations** link to enter a note for the element. For more information, see Element annotations.
   - Click the **Security** link to set up custom access permissions for the element. For more information, see PI AF object security.
6. To save your work, press CTRL-S or click **Check In**.

Find where an element is used

Use the **Find** links on the **General** tab for a selected element to locate where and how it is used in the asset model.

Procedure

1. Select an element in the Elements browser.
2. Choose from the following options:

To find ...	Do this ...
Parent elements	Click the **Parents** link. The Parents of *Element* window opens.
Child elements	a. Click the **Children** link. b. In the Element Search window, enter element name text in the search field. To refine your search, you can enter additional criteria in the **Criteria** section. c. Click **Search**. Matching elements are displayed in the **Results** section.
Event frames connected to the element	Click the **Event Frames** link. ◦ If a match is found, the Find Event Frames for *Element* displays a list of all event frames that reference the element. ◦ If no match is found, a `No Event Frames Found` message is displayed.

Deletion of elements

When you delete an element in PI AF 2.4 and later, PI AF automatically deletes the notifications and analyses that target that element. If you wish to repurpose a notification that is attached to an element that you plan to remove, ensure you do so before you remove the element.

Alternatively, you can change the default behavior so that PI AF does not automatically delete notifications and analyses when you delete the targeted element.

You use the **afdiag** command line utility with the **/EnablePropagationOfTargetDeletion** parameter:

```
afdiag /EnablePropagationOfTargetDeletion-
```

You should be aware that disabling the setting can cause problems, such as:

- Notifications are left open (and remain active)
- New notifications are not created
- Expected emails are not sent

Attribute-value reset to original properties

When you have created elements based on templates in a test environment and later want to transfer your work to a production environment, you can reset all attributes for an element to the original properties as defined by a template. You can right-click an attribute for an element and click **Reset to Template**.

For example, suppose you configure a PI AF attribute to use the name of PI Data Archive as part of the attribute's configuration in a test environment. You might want this to reflect the new name for PI Data Archive when you transfer your work to a production environment.

Convert elements to element templates

OSIsoft recommends that elements be based on templates. If you created elements and attributes and did not originally base them on a template, you can later convert them to a template.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, locate the element that you want to convert to a template.
3. Right-click the element and click **Convert** > **Convert to Template**.

   A template named ***ElementName*Template** is created and displayed in the **Template** field on the **General** tab.
4. Optional. To rename the template, click **Library** in the navigator.
   a. In the Library browser, locate the new template in the **Element Templates** branch.
   b. Right-click the template and click **Rename**.
   c. Modify the default name as needed. The new name must be unique across all templates.

Change element templates

You can change the template on which an existing element is based, or add a template to an element that is not currently based on a template.

Before you start

Exercise caution when you change an element template, since there can be unintended consequences. For example, attributes could be deleted if they were defined by the old template and are not present in the new template. Be sure that attributes with the same name in both the old and new template have the same data type.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, right-click the element that you want to change and click **Convert > Change Template**.
3. In the Choose Element Template window, select a template from the **Element Template** list.
   a. Optional. To display only templates in a specific category, select a category from the **Templates of category** list.
4. Click **OK**.

Storage of application-specific information

Extended properties are properties that other applications define on PI AF objects. For example, PI WebParts stores Icon and URL in a PI AF element's extended properties. Applications typically make use of the information stored in the Extended Properties window programmatically with AF SDK.

An **Extended Properties** link is displayed on the **General** tab of many object types in PI System Explorer. This link can be used to configure one or more additional properties for an object.

> **Note:**
> Users do not need to use this advanced PI AF feature.

Create extended properties

In general, client applications such as PI WebParts and PI ProcessBook use the Extended Properties feature to write properties, and users do not need to use this advanced feature.

Procedure

1. Click the **Extended Properties** link.
2. In the Extended Properties window, choose from the following actions:
   - To create the first extended property, click the **New Extended Property** link.
   - To create an additional extended property, right-click in the **Name** column and select **New Extended Property**.
3. In the **Name** field, delete the default text and enter the extended property name.
4. Click in the **Value** field and choose from the following options:
   - Enter a string. To visualize the text you are entering, press F2 and type in the Text Visualizer window.
   - Click T and select from the list of **Basic Types**, **Array Types**, and **Objects**.
     - If you choose an object such as **Attribute** or **Element**, click ... and locate the desired object.
     - If you choose **File**, click ▸ and select **Upload** to locate the desired file.
5. Click **Close**.

Association of data with PI AF assets

To associate data with an asset, you create attributes on the element that represents that asset.

Attributes can hold simple values that represent fixed information, such as the diameter of a tank. They can also hold values from PI points, be derived from formulas, or hold data from outside the PI System through the use of PI AF tables.

Enumeration sets

If you need an attribute to hold only predefined values, you can use an enumeration set. An enumeration set maps an ordered set of user-defined constant values to a set of strings. You can use the strings to provide brief, descriptive text to use within your PI AF model.

Videos

For information on how to create element attributes, watch this video:

https://www.youtube.com/watch?v=pLmIPOOIcEw&rel=0

For information on how to create child attributes, watch this video:

https://www.youtube.com/watch?v=BiXA13ptVTM&rel=0

Topics in this section

- Attribute templates
- Enumeration sets

Attribute templates

Attribute templates are associated with element templates. Just as an element template represents a type of asset, an attribute template represents a type of data configuration. When you create an instance of the element template, that element contains an attribute for each attribute template. These attributes inherit all properties configured on the attribute template.

Rather than create attributes on each element, OSIsoft recommends that you create attribute templates in an element template. Whenever you create elements based on that template, PI AF automatically creates the attributes for you. You still need to give each attribute a value.

Modification of attribute descriptions created from attribute templates

Beginning with 2017 R2, you can modify the descriptions of attributes that are created from a template. You may find this helpful when differentiating between attributes that use the same attribute template. For example, in a series of pumps, each pump has a `Flowrate` attribute with a PI point data reference. The template description for `Flowrate` is simply `Pump flowrate`. To make that description unique for each instance of the pump, you could use PI Builder to load the **`descriptor`** attribute for each pump flowrate PI point and incorporate that value into each `Flowrate` description, thereby customizing it.

To modify descriptions created from an attribute template, you must use a PI AF client and server that are running PI AF 2017 R2 or later. Older clients that connect to a PI AF server running 2017 R2 or later continue to see only the description from the attribute template.

> **Note:**
> Because extra memory is used to store descriptions for attributes that have been overridden, loading them into client applications can take a little longer.

Video

For information on how to create attribute templates, watch this video:

https://www.youtube.com/watch?v=dCx5_Aw5x24&rel=0

Attribute properties

You need to set up the properties for each element attribute. After you assign properties in an attribute template, users can only modify the **Exclude** property on individual attributes.

Topics in this section

- Configuration Item attribute property
- Excluded attribute property
- Hidden attribute property
- Indexed attribute property
- Manual Data Entry attribute property

Configuration Item attribute property

You assign the **Configuration Item** property to an attribute that represents inherent properties of an asset. Any attribute that has a constant value can be a configuration item. Because the attribute value of a configuration item is presumed to be constant, PI System Explorer automatically checks out the attribute when you change the attribute value. To commit the change, you need to check it in. In PI System Explorer, configuration attributes are marked with a pencil icon (✎).

Attributes with values that change, such as PI point data references, or formula data references, should not be configuration attributes. When you change the value of a non-configuration attribute, PI System Explorer does not check out the attribute. For example, suppose you have an element representing a tank. The element has the following attributes:

- ✎ manufacturer
- ✎ photo (file)
- ✎ serial number
- ✎ maximum temperature rating
- ✎ maximum volume
- temperature (PI point data reference)
- volume (PI point data reference)

  The first five attributes would typically be configuration items, since these values will not change unless you change the equipment. The last two attributes would not be configuration items, since the values change all the time.

Excluded attribute property

You exclude an attribute in situations where not all attributes in an element template apply. For example, suppose you have five attribute templates defined in an element template named `Meter`, which you use to create elements representing several different types of meter in use at your facility. In some kinds of meter (`meterA`), only three attributes from the `Meter` template are applicable, whereas in the remainder (`meterB`) all five attributes apply. In meters of type A, you should select the **Excluded** property for the two attributes that are inapplicable. For all practical purposes, those two attributes "seem" nonexistent to most PI AF client applications since searches retrieve the digital state value `No Result`.

> **Note:**
> An excluded attribute cannot be used to return elements and event frames that are referenced by an excluded attribute value.

When the **Excluded** property is assigned in an attribute template, the property is indicated by beside the attribute name if the **Show Excluded Attributes** option is enabled. Attributes that are derived from that template appear on an element **Attributes** tab with in the Template column and **Excluded** in the Value column. You can toggle the display of such attributes with the **Hide Excluded Attributes** and **Show Excluded Attributes** command on the **Attributes** tab context menu. Note that when **Show Excluded Attributes** is enabled, users can change the **Excluded** property setting for an attribute from the setting it is assigned in an attribute template.

Alternatively, you can clear the **Show Excluded Attributes** option in **Tools** > **Options** to hide excluded attributes everywhere in PI System Explorer. When attributes are hidden, the `Excluded attributes are hidden` message is displayed on the **Attributes** tab.

Hidden attribute property

A hidden attribute is one that cannot be retrieved in searches from PI AF client applications such as PI Vision. This property is useful if an attribute is being used to hold an intermediate result, such as a table lookup result that can then be retrieved by a PI point data reference, or is being used solely to populate a PI point name in a substitution parameter.

It can also be useful to set an attribute property to **Hidden** in a template when configuration has not been fully completed in the element itself. For example, elements are being created from a template with a PI point data reference, but because some instrumentation is missing, PI points do not yet exist for all the elements. By setting the attributes for the missing instrumentation to **Hidden**, a PI AF client application is prevented from obtaining an error result from a search.

When the **Hidden** property is assigned to an attribute in an attribute template, the property is indicated by in the Hidden column beside the attribute name. The same icon is displayed next to the attribute name on the **Attributes** tab when the template is assigned to an element.

Indexed attribute property

You index an attribute to optimize it for fast search results and fast value retrieval. Indexes only improve performance for attributes whose values are stored in the PI AF database. This means that you do not gain any benefit from indexing attributes that obtain their values from data references, as is the case with PI points and linked tables. For event frames, however, when

attribute values are captured they are written to the PI AF server and as a result can benefit from indexing.

> **Note:**
> Only the first 40 characters of indexed attributes with a **String** value type are used in searches.

When you index an attribute, both the attribute and the attribute value are returned more quickly in a search. However, indexed attributes increase the table size in your PI AF database and they create some CPU load on the SQL server. Indexed attributes are marked with the **i** icon.

You typically index an attribute that you think users are likely to search for frequently. For example, if you know that you want to locate assets by searching for a serial number, you should index the serial number attribute.

Manual Data Entry attribute property

You flag an attribute for manual data entry to enable client applications (such as PI Vision and PI Manual Logger) to guide users to enter data for that attribute. For example, in a future release of PI Manual Logger, this property will identify tags in a tour that operators should collect manually, in contrast to tags that are collected automatically via an interface or connector.

To determine which attributes have this property, add a Manual Data Entry column to an attribute grid in the **Elements** viewer, where attributes with this property display as **True**.

Attribute traits

Attribute traits identify commonly held characteristics or related behaviors of a parent attribute. PI AF supports attribute traits for limit conditions, forecasted values, geolocation information, reason codes, asset health, and analysis start triggers. Users and client applications like PI Vision can then find, display, and analyze such related attributes more effectively.

To set and access attribute traits, both the PI AF client and server need to be running PI AF 2016 or later.

Limit attribute traits

You use limit attribute traits to identify the expected range of a process variable or a measurement attribute. Limit attribute traits are often set elsewhere (for example, on a control system), or represent inherent properties of the parent attribute, such as temperature. A typical usage would be to set up conditions for an alarm. PI Vision, for example, uses most limit attributes as input when configuring multi-state behavior for a visual alarm.

Limit attribute traits are child attributes of the measurement they describe and have the same value type and UOM as the parent. The only exception is when the value of a limit attribute trait is obtained from a data reference, in which case the source UOM of a defined data reference can be different. You can also only set limit attribute traits for parent attributes that are numeric. Note that you can only assign a specific limit attribute trait once to a parent attribute.

The following limit traits are defined:

Limit attribute trait	Description
Minimum	Indicates the very lowest possible measurement value or process output.
LoLo	Indicates a very low measurement value or process output, typically an abnormal one that initiates an alarm.
Lo	Indicates a low measurement value or process output, typically one that initiates a warning.
Target	Indicates the aimed-for measurement value or process output.
Hi	Indicates a high measurement value or process output, typically one that initiates a warning.
HiHi	Indicates a very high measurement value or process output, typically an abnormal one that initiates an alarm.
Maximum	Indicates the very highest possible measurement value or process output.

Video

For information on how to set limit attribute traits, watch this video:

https://www.youtube.com/watch?v=jO2dVYsBvds&rel=0

Forecast attribute traits

In general, you use forecast attribute traits as predicted values to compare with the actual value of the parent attribute. Typically, such attribute traits represent future PI points but that is not a requirement. Client applications can very easily relate these attributes and trend them together.

Forecast attribute traits are child attributes of the attribute they describe and have the same value type and UOM as the parent. The only exception is when the value of a forecast attribute trait is obtained from a data reference, in which case the source UOM of a defined data reference can be different. You can also create forecast attribute traits for parent attributes that are of any value type. Note that you can have multiple forecast attribute traits defined for a parent attribute.

Location attribute traits

You use location attributes to define longitude, latitude, and altitude information for an asset. You can use this information to identify the location of the asset on a map.

You can create location attribute traits on an element, an event frame, a model, or on their child attributes. However, you can create only one of each attribute trait per element, event frame, or model. The UOM for the latitude and longitude attribute traits should be degree (the default) or at least an angle, whereas the altitude attribute trait can be meter (the default) or another UOM in the Length class. The value type for all location attribute traits must be numeric (double).

Reason attribute traits

You use reason attribute traits on event frames and transfers to enable users to select a reason code for excursions, downtime, and other events. You can set a single reason attribute trait on an event frame or transfer template or an event frame or transfer instance. The reason attribute trait must be an enumeration set that is previously defined, or a system enumeration set delivered with PI AF.

Health attribute traits

You use health attribute traits on elements and models to enable users to set a numeric health score and a health status (for example, healthy, out of service, in maintenance, warning, or error). The HealthStatus attribute trait uses values from the **Health Status** enumeration set, which is delivered with PI AF. Administrators can modify the **Health Status** enumeration set as required.

Analysis start-trigger attribute traits

When users configure analytics to generate event frames, they can optionally elect to store the name of the start trigger in the value of an attribute (string) and mark that attribute with the analysis start trigger trait. This enables clients like PI Vision to indicate the start trigger that created that particular event frame. For more information, see Event frame generation analyses.

Topics in this section

- Set limit attribute traits
- Set forecast attribute traits
- Set location attribute traits
- Set reason attribute traits
- Set health attribute traits

Set limit attribute traits

Set up to seven limit traits for an attribute template or an element attribute to enable users to define the expected range of a process variable or a measurement.

Procedure

1. Choose from the following actions:

To set a limit trait for an ...	Do this ...
Attribute template	a. In the Library browser, select an element template. b. In the viewer, click the **Attribute Templates** tab.
Element attribute	a. In the Elements browser, select an element. b. In the viewer, click the **Attributes** tab.

2. Select the attribute for which you want to set one or more limit traits.
3. Choose one of the following actions:
   - Right-click and click **Limits**.
   - At the bottom of the configuration area, click the **Limits** link.
4. In the Limits window, select the check box beside the limit trait you want to set.
5. In **Attribute**, choose one of the following actions:

- Leave the default attribute name as is.
- Replace the default attribute name with a revised name of your choice.
- If a child attribute already exists that you want to assign to the limit trait, select it from the drop-down list.

> **Note:**
> Limit attribute traits that have already been defined are displayed in bold typeface.

6. Choose from the following actions:

To set a ...	Do this ...
Value for the limit trait	In **Value**, enter a limit value.
Value for the limit trait from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. **Note:** The source UOM of a data reference can be different from that of the parent attribute. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

7. Repeat steps 4 to 6 to set values for other limit traits.
8. Click **OK** when your list of limit traits for the selected parent attribute is complete.

Set forecast attribute traits

Set one or more forecast traits for an attribute template or an element attribute to enable users and client applications to compare predicted values with actual values and plot a trend.

Procedure

1. Choose from the following actions:

To set a forecast trait for an ...	Do this ...
Attribute template	a. In the Library browser, select an element template. b. In the viewer, click the **Attribute Templates** tab.
Element attribute	a. In the Elements browser, select an element. b. In the viewer, click the **Attributes** tab.

2. Select the attribute for which you want to set one or more forecast traits.
3. Choose one of the following actions:
   - Right-click and click **Forecasts**.
   - At the bottom of the configuration area, click the **Forecasts** link.
4. In the Forecasts window, click the **New Forecast** button or link.

5. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as is.
   - Replace the default attribute name with a revised name of your choice.
   - If a child attribute already exists that you want to assign to the forecast trait, select it from the drop-down list.

   Note:
   Forecast attribute traits that have already been defined are displayed in bold typeface.

6. Choose from the following actions:

To set a ...	Do this ...
Value for the forecast trait	In **Value**, enter a forecast value.
Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. **Note:** The source UOM of a data reference can be different from that of the parent attribute. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

7. You can set as many forecasted values as necessary. Repeat steps 4 to 6.
8. Click **OK** when your list of forecast attributes for the selected parent attribute is complete.

Set location attribute traits

Set one or more location traits on an element or event frame template or on an element or event frame to enable users to identify the location of an asset on a map. You can also set a location for a model.

Before you start

You can only set one set of location attribute traits per element, model, or event frame.

Procedure

1. Choose from the following actions:

To set a location trait on ...	Do this ...
Element or model template	a. In the Library browser, select an element or model template. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.

Event frame template	a. In the Library browser, select an event frame template. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.
Element or model	a. In the Elements browser, select an element or model. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.
Event frame	a. In the Event Frames browser, select an event frame. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.

Note:
You can also configure location traits as a child attribute template to an existing attribute template, or as a child attribute to an existing attribute:

- On a template, right-click an attribute template and click **Location of *Element/Event Frame/Model* Template**.
- On an element, event frame, or model, right-click an attribute and click **Location of *Element/Event Frame/Model***.

2. In the Location window, select the check box beside the trait you want to apply, or select the check box in the column header to select all traits.
3. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as is. The default attribute name displays the path relative to the element.
   - Replace the default attribute name with a name of your choice. Ensure that the full path to the element is retained.
   - If a child attribute already exists that you want to assign to the location trait, select it from the list.

Note:
Location attribute traits that have already been defined are displayed in bold typeface.

4. Choose from the following actions:

To set a ...	Do this ...
Value for the location trait	In **Value**, enter a location value.

Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when your list of location attributes for the selected element is complete.

Set reason attribute traits

Use reason attribute traits on event frames and transfers to enable users to select a reason code for excursions, downtime, and other events.

Before you start

You can only set one reason attribute trait per event frame or transfer.

Procedure

1. To designate an attribute as a reason attribute trait for an event frame or transfer, choose from the following actions:

To set a reason attribute trait for ...	Do this ...
Event frame template	a. In the Library browser, select an event frame template. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Transfer template	a. In the Library browser, select a transfer template. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Event frame	a. In the Event Frames browser, select an event frame. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Transfer	a. In the Event Frames browser, select a transfer. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.

Attribute template for an event frame template	a. In the Library browser, select an event frame template. b. Click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute template and click **Reason for Event Frame Template**. ▪ Click **New Attribute Template**, right-click and click **Reason for Event Frame Template**.
Attribute template for a transfer template	a. In the Library browser, select a transfer template. b. Click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute template and click **Reason for Transfer Template**. ▪ Click **New Attribute Template**, right-click and click **Reason for Transfer Template**.
Event frame or transfer attribute	If the **Allow Extensions** check box has been selected for an event frame or a transfer template, and no attribute template has previously been designated as a reason attribute trait, you can either create a new attribute and establish it as the reason attribute trait, or choose an existing attribute. a. In the Event Frames or Transfer browser, select an event frame or transfer. b. Click the **Attributes** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute and click **Reason for Event Frame** or **Reason for Transfer**. ▪ Right-click below the attribute table and click **Reason for Event Frame** or **Reason for Transfer**.

2. In the Reason window, select the check box beside the Reason trait you want to apply.
3. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as **Reason**.
   - If an attribute already exists that you want to designate as the reason trait, select it from the list.

   Note:
   Reason traits that have already been defined are displayed in bold typeface.
4. Choose from the following actions:

To set a ...	Do this ...
Default value for the reason trait	a. In **Value Type**, select **Enumeration Value** > **Enumeration Sets** and select an enumeration set to use for the reason trait. The default is the built-in **System** enumeration set. b. In **Default Value**, select the default reason trait value from the selected enumeration set.
Default value from a data reference **Note:** The data reference must be to a digital states tag.	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when the assignment of a default reason trait for the selected event frame or transfer is complete.

Set health attribute traits

Set health attribute traits for an element template or for an element to enable users to establish the health of an asset. You can also set health attribute traits for a model template or model.

Before you start

You can only set one set of health attribute traits per element or model.

Procedure

1. Choose from the following actions:

To set a health trait for ...	Do this ...
Element or model template	a. In the Library browser, select an element or model template. b. Choose one of the following actions: ▪ Right-click and click **Health**. ▪ On the **General** tab, click the **Health** link.
Element or model	a. In the Elements browser, select an element or model. b. Choose one of the following actions: ▪ Right-click and click **Health**. ▪ On the **General** tab, click the **Health** link.

Attribute template	a. In the Library browser, select an element or model template. b. In the viewer, click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click under the grid and click **Health of Element Template**. ▪ Click **New Attribute Template** and from the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.
Attribute	a. In the Elements browser, select an element or model. b. In the viewer, click the **Attributes** tab. c. Choose one of the following actions: ▪ Right-click under the grid and click **Health of Element**. ▪ Click **New Attribute** and from the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.

Note:
You can also configure health traits as a child attribute template to an existing attribute template, or as a child attribute to an existing attribute by right-clicking and clicking **Properties**. From the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.

2. In the Health window, select the check box beside the trait you want to apply, or select the check box in the column header to select all traits.
3. In the **Attribute** column of the Health window, choose one of the following actions:
   - Leave the default attribute name as is. The default attribute name displays the path relative to the element.
   - Replace the default attribute name with a name of your choice. Ensure that the full path to the element is retained.
   - If a child attribute already exists that you want to assign to the health trait, select it from the list.

   Note:
   Health attribute traits that have already been defined are displayed in bold typeface.
4. Choose from the following actions:

To set a ...	Do this ...
Value for the HealthScore trait	In **Default Value** (for a template) or **Value** (for an element) , enter a numeric value.

Value for the HealthStatus trait	In **Default Value**, select a Health Status enumeration value: ◦ **Healthy** ◦ **Out of Service** ◦ **In Maintenance** ◦ **Warning** ◦ **Error** ◦ Other Health Status enumeration set values added by an administrator
Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when your list of health attributes for the selected element (or model) is complete.

Control the display of attribute and attribute template values

Prior to PI AF 2018, PI System Explorer rendered numeric data types in displays with high levels of precision. For example, the Single and Double data types have a precision of seven and fifteen respectively. Such high levels of precision are not always useful in real-time situations, or for PI AF client products: for example, a temperature gauge may only display a precision of three digits, so process engineers are unlikely to require a greater degree of precision for mathematical calculations.

Beginning with PI AF 2018, the **Use DisplayDigits for Attribute and Attribute Template values** system option is enabled and ensures that the default setting of the DisplayDigits property is used to display attribute values: -5, or a total of five digits overall. This is the same default setting used for PI points on PI Data Archive.

When creating attribute templates, you can now control the number of digits you want to see for individual attributes in the **Display Digits** field:

- You can enter a negative number up to -20 to indicate the total number of digits to display.
- You can enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

When you are editing an attribute value, the actual number is displayed rather than the truncated number established by the **Display Digits** field.

For attributes that are not based on a template, the precision of PI point data references is obtained from the PI point itself.

Note:
On upgrades, existing attributes configured with a PI Point data reference are not updated to obtain their DisplayDigits value from the PI point: they are set to -5.

Create attribute templates

Use attribute templates to create a standard set of attributes that are associated with element, event frame, model, and transfer templates.

Procedure

1. In the Library browser, select a template.
2. To create a new attribute template, choose one of the following actions:
   - On the toolbar, click **New Attribute Template**.
   - Right-click the template in the browser and click **New** > **New Attribute Template**.
3. In the **Name** and **Description** fields, enter a unique name and a description for the attribute template.

   Note:
   When you need to search for an attribute, you can search on up to 440 characters of a description.

4. Set the configuration fields, as needed.

Field	Description
Properties	Select one or more of the following properties, as needed. ◦ **Configuration Item** ◦ **Excluded** ◦ **Hidden** ◦ **Indexed** ◦ **Manual Data Entry** For more information on these properties, see Attribute properties. You can set up the current attribute template to be a location attribute trait for a parent object. Select one of: ◦ **Location** > **Longitude** ◦ **Location** > **Latitude** ◦ **Location** > **Altitude** When the parent object is an event frame or transfer, you can set up the current attribute template to be a reason attribute trait. For more information, see Set reason attribute traits. When the parent object is an element or model, you can set up the current attribute template to be an asset health attribute trait. For more information, see Set health attribute traits. **Note:** When you configure a child attribute template, you can also select limit and forecast attribute traits, depending on the value type of the parent attribute template. For more information, see Attribute traits.
Categories	Click and select one or more previously defined categories from the Categories window. You can also create a new category. For more information, see Categorization of objects.

Default UOM	Select a UOM from the list. For more information on UOMs, see Units of measure in PI AF. **Note:** Although a user can change the UOM that is displayed for an attribute in PI System Explorer, the UOM that is defined in the template does not change.

5. Configure the type of value that the attribute holds.
   - For attributes with constant values, set the attribute **Value Type**. Constant attribute values can be numbers, strings, files, date-times, Boolean, URLs, arrays, and more. For more information, see Define constant values for attribute templates.

     Note:
     Do not set a default UOM for integer value types if searches by client applications, such as PI Vision, will be performed on the value or a UOM conversion is involved. For example, an integer value type with a default UOM is not compatible with the UOM groups feature introduced in PI AF 2017 R2. Use the **Single** or **Double** value type instead.

     Note:
     While allowed, the **Anything** value type is not recommended. When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, formulas, and analytics cannot function with attributes that have no value type.

   - For PI point values, PI point arrays, formulas, and table data (including data from relational databases), click **Settings** to configure a data reference. For more information, see Configuration of data references.
6. Optional. Enter a default value for the attribute in the **Default Value** field.
7. Optional. When the **Use DisplayDigits for Attribute and Attribute Template values** system option is selected, you can override the default DisplayDigits property value of -5 and specify the number of digits to display in the **Display Digits** field. Only numeric values in the **Value** field are affected.
   - Enter a negative number up to -20 to indicate the total number of digits to display.
   - Enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

   For more information, see Control the display of attribute and attribute template values and PI System Explorer customization options.
8. To save your work, press CTRL-S or click **Check In**.

After you finish

When you create an attribute based on the template, you need to set the value:

- For constant values, add the value to the attribute directly.
- For data references, you need to create the instance of the data reference. To do this, right-click on the element in the browser and choose **Create or Update Data Reference**. For more information, see Attribute-value updates from PI point data references.

> **Note:**
> The way you configure an attribute can sometimes restrict how the data can be viewed in client applications, such as PI ProcessBook or PI Vision. See Restrictions on viewing time series data for more information.

PI AF data types

The following table contains definitions of the data types in use in PI AF.

Data type	PI Data Archive equivalent	Description	Range
`Boolean`	None	A data type that tracks true or false conditions	`True` or `False` A `Boolean` attribute can be stored into a PI point of type `String` and any of the `Integer` tags, but `Digital` is the preferred representation.
`Byte`	None	An 8-bit unsigned data type	0 to 255 These values can be represented in a larger PI data type such as `Int16`. However, if PI Data Archive contains data outside the byte range, an error is returned when retrieved through PI AF.
`DateTime`	`Timestamp`	Date and time stored internally in 64 bits	• If stored in PI Data Archive: 1 Jan 1970 to 19 Jan 2038, with a precision of 15 microseconds • If not stored in PI Data Archive: 1 Jan 0001 to 31 Dec 9999, with a precision of 100 nanoseconds
`Double`	`Float64`	A 64-bit floating point number	±5.0e−324 to ±1.7e308, 15 digits of precision
`GUID`	None	A unique 128-bit data type used for the global identification of objects	00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff The above values can be represented as a `String` PI data type.
`Int16`	`Int16`	A 16-bit signed integer	-32,768 to 32,767
`Int32`	`Int32`	A 32-bit signed integer	-2,147,483,648 to 2,147,483,647
`Int64`	None	A 64-bit signed integer	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 There is no equivalent of an `Int64` in PI Data Archive. For non-numerical representations (such as an ID number), a string may be used. For numerical representations, `Float64` provides some equivalence but will be subject to rounding issues with very large (positive or negative) numbers.
`Single`	`Float32`	A 32-bit floating point number	±1.5e−45 to ±3.4e38, 7 digits of precision

Data type	PI Data Archive equivalent	Description	Range
`String`	`String`	A sequence of multiple characters	PI Data Archive strings limited to 976 characters, and are not unicode

Define constant values for attribute templates

You can assign constant values directly on an attribute template. Although you cannot set actual attribute values in attribute templates, you can define default values.

Procedure

1. In the viewer, use the **Value Type** field to select the data type of the attribute. There are four groups of value types:
   - **Basic Types**

     A data type supported by PI AF, as described in PI AF data types.
   - **Array Types**

     An array contains any of the Basic types as elements.
   - **Enumeration Sets**

     An enumeration set is a list of user-defined constant values. You typically use an enumeration set to define a list of predefined values for an attribute template. When you configure attributes based on that template, you can select a value from the list. Use the **Enumeration Sets** option to select any of the enumeration sets that have already been created in your PI AF database.
   - **Objects**

     An object can be another PI AF attribute, element, or an operating system file.
2. Click in the **Default Value** field.

   The format of the **Default Value** field changes according to the **Value Type**.
3. Configure the value for the attribute, as described in the appropriate procedure for the attribute data type.

Topics in this section

- Configure values for Basic data types
- Configure values for Array data types
- Configure Enumeration Set values
- Configure values for Object data types

Configure values for Basic data types

After you have set **Value Type** to one of the Basic data types, you can define the attribute's value.

Procedure

1. Click in the **Default Value** field. The format of the field changes according to the value type. See PI AF data types.
2. Enter the attribute's value in the **Default Value** field. PI System Explorer resolves the data you enter and when you check in your changes, it sends the resulting value to PI AF. For example, for **DateTime**, it sends the resulting date and time.

Examples

If the value type is **DateTime**, you can type the time in any string format that is supported by PI AF (including PI Time formats) or the .NET DateTime object. Some examples of valid entries:

```
*-5d
May 12, 2009
07 06 2010  10:00:00 AM
09 14 2008  14:00:00
```

To create an attribute with a link as a value, select the value type **String** and enter the URL as the attribute value. Strings that are recognized as absolute URL paths will be displayed as a link. For example, strings starting with `http://`, `ftp://`, `file://` and `www.` are recognized as links, whereas strings starting with `C:\`, and `abc.com` are not. In PI System Explorer, links appear underlined and have an associated tooltip.

Configure values for Array data types

After you have set **Value Type** to one of the Array data types, you can define the attribute's value.

Procedure

1. Click in the **Default Value** field.
2. Click ... beside the **Default Value** field.
3. In the Array window, define your array. The **Default Value** field is set to the appropriate format for the data type you selected for your array elements. See PI AF data types.
4. Click **OK**. PI System Explorer resolves the data you have entered and when you check in your changes, it sends the resulting array values to PI AF.

Configure Enumeration Set values

When you select an enumeration set for the **Value Type**, you can pick the value for an attribute from that set's predefined list of constant values. For more information on enumeration sets, see Enumeration sets.

Procedure

1. Click in the **Default Value** field. The **Default Value** field becomes a drop-down with values from the selected enumeration set.
2. Select a value from the list of predefined constants. When you check in your changes, PI System Explorer sends the resulting value to PI AF.

Configure values for Object data types

After you have set **Value Type** to one of the Object data types, you then define the attribute's value.

Procedure

1. Click in the **Default Value** field. The action button beside the field changes according to the selected value type.

Value Type	Action
<Anything>	While allowed, the Anything value type is not recommended. **Note:** When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, Formulas, and Analytics cannot function with attributes that have no value type.
Attribute	Click ... to open the Attribute Search window. Enter search criteria, as described in Search for attributes on elements.
Element	Click ... to open the Element Browser window, where you can locate and select an element.
File	Click ▸ and select **Upload**. Locate and select a file in the Open window. For acceptable types of attachment files, see `/FileExtensions` in AFDiag utility parameters. **Caution:** Before you upload a file, run an anti-virus check and ensure that the file size does not exceed your storage space. Some file types can pose a security risk, and a warning message will be displayed, asking if you are sure you want to continue. The list of file types that PI System Explorer considers unsafe can change. As a guideline, look at Microsoft's list of file types blocked by Outlook.

2. Click **OK**. When you check in your changes, PI System Explorer sends the resulting value to PI AF.

Add attribute extensions to template-based elements

In some situations, you might want to add another attribute to an existing element. If the element is based on a template that allows extensions, you can define additional attributes in the element itself independently of the template.

Before you start

The **Allow Extensions** check box on the **General** tab for the element template upon which the attribute is based must be checked.

Procedure

1. In the Elements browser, select the element to which an attribute will be added.
2. Choose from the following actions.
   - In the toolbar, click **New Attribute**.
   - Right-click the element in the browser and click **New** > **New Attribute**.
   - Select the **Attributes** tab in the viewer, right-click and click **New Attribute** from the context menu.
3. In the **Name** and **Description** fields, enter a unique name and a description for the attribute template.

   Note:
   When you need to search for an attribute, you can search on up to 440 characters of a description.

4. Configure the type of value that the attribute holds:
   - For attributes with constant values, set the attribute **Value Type**. This setting determines what you can enter in the **Value** field.

     Constant attribute values can be numbers, strings, files, date-times, Boolean, URLs, arrays, and more.

     Note:
     While allowed, the **Anything** value type is not recommended. When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, formulas, and analytics cannot function with attributes that have no value type.

     Optional. Enter or modify the default value for the attribute in the **Value** field.
   - For formulas, PI point values, PI point arrays, and table data (including data from relational databases) click **Settings** to configure a data reference.

     See Configuration of data references for information about configuring different types of data reference.
5. Set other configuration fields, as needed.

Field	Description
Properties	Select one or more of the following properties, as needed. ◦ **Configuration Item** ◦ **Excluded** ◦ **Hidden** ◦ **Indexed** ◦ **Manual Data Entry** For more information, see Attribute properties.

Categories	Click and select one or more previously defined categories from the Categories window. You can also create a new category. For more information, see Categorization of objects.
Default UOM	Select a UOM from the drop-down list. For more information, see Units of measure in PI AF. **Note:** You can change the UOM that is displayed for the attribute in PI System Explorer; however, the UOM defined in the template is not changed.

6. To save your work, press CTRL-S or click **Check In**.

Enumeration sets

You typically use enumeration sets to establish predefined values for attribute templates. When you configure element attributes based on those templates, you can then have users select those values from pre-populated lists rather than typing values manually. This helps ensure you have consistent nomenclature throughout your database.

Hierarchical enumeration values

Beginning with PI AF 2017 R2, you can nest enumeration values in a hierarchy. For example, suppose you have an enumeration set of pump manufacturers, with child enumeration values for pump types. Within each pump type you could also create child enumeration values for different models. You can create as many levels as you need.

Value	Name	Description
0	Manufacturer A	
5	Manufacturer A|Air operated diaphr...	
10	Manufacturer A|Booster	
15	Manufacturer A|End suction	
16	Manufacturer A|End suction|Series I	Temp under 80º c or 176º F
17	Manufacturer A|End suction|Series II	Hot water pump temp under 120º c or 248º F
18	Manufacturer A|End suction|Series III	Chemical pump temp under 120º c or 248º F
20	Manufacturer A|Magnetic drive	
25	Manufacturer A|Multistage	
30	Manufacturer A|Self-priming	
35	Manufacturer A|Vertical sump	
40	Manufacturer B	
45	Manufacturer B|Chopper	
50	Manufacturer B|Dry pit	
55	Manufacturer B|Grinder	
60	Manufacturer B|Vertical well	

Sample hierarchy of enumeration values

When you configure the `Pump` attribute template, you can simply select the `Pump Manufacturer` enumeration set as the **Value Type**, and thereby enable users to select predefined pump types and model numbers.

Sample enumeration set in attribute template

Sort order

Beginning with PI AF 2017 R2, wherever predefined values defined by enumeration sets are displayed, users can select **Sort By** > **Sort By Name** to sort those values by name, or **Sort By** > **Sort By Value** to sort by enumeration set value.

Video

For information on how to create enumeration sets, watch this video:

https://www.youtube.com/watch?v=nH7zvsb8kBE&rel=0

Manage enumeration sets

You use enumeration sets in attribute templates whenever you want to be able to select from lists of predefined values when you are defining element attributes.

Procedure

1. In the navigator, click **Library**.
2. Choose from the following actions.

To ...	Do this ...
Create a new enumeration set	a. Choose one of the following actions. ▪ On the toolbar, click **New Enumeration Set**. ▪ In the browser, right-click the **Enumeration Sets** collection and click **New Enumeration Set**. b. In the **Name** field, enter a unique name for the enumeration set. c. Optional. In the **Description** field, enter a description for the enumeration set. d. Optional. To use hexadecimal values, select the **Hexadecimal** check box. e. Optional. If you want to configure access permissions for the new enumeration set that are different from those inherited from the **Enumeration Sets** collection, click the **Security** link. For more information, see Configure security for objects.
Create a new enumeration value	a. Choose one of the following actions. ▪ In the browser, click an existing enumeration set. In the viewer, right-click in any **Value** field and click **New Enumeration Value**. ▪ In the browser, right-click an existing enumeration set and click **New Enumeration Value**. The value is a unique numeric value associated with the enumeration and provides a quicker, less memory-intensive representation of an enumeration's value. b. In the **Value** field of the Enumeration Value Properties window, either enter a unique number manually, or click to increase or lower the value. c. In the **Name** field, enter a unique string that describes the condition or state being represented. This string is used as the displayed value when an enumeration set is selected as the value type for an attribute. d. Optional. In the **Description** field, enter a description for the enumeration value.

Create a new child enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click an existing value to which you want to add a child enumeration value, and click **New Child Enumeration Value**. c. In the Enumeration Value Properties window, verify that the value selected in the **Parent** field is correct. Otherwise, select a different parent from the list. d. In the **Value** field, either enter a unique number manually, or click to increase or lower the value. e. In the **Name** field, enter a unique string that describes the condition or state being represented. This string is used as the displayed value when an enumeration set is selected as the value type for an attribute. f. Optional. In the **Description** field, enter a description for the enumeration value.
Rename an enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click the existing value that you want to rename and click **Properties**. c. In the **Name** field of the Enumeration Value Properties window, revise the name string as needed but ensure it remains unique. If the name is being used as a parent enumeration value, note that the name change will update the parent portion of all child enumeration value strings.
Renumber enumeration values	See Renumber enumeration values.
Delete an enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click the existing value that you want to delete and click **Delete**. c. Click **Yes** in response to the Delete warning. Note that you will invalidate objects where the value is currently being used.

3. On the toolbar, click **Check In**.

Renumber enumeration values

You can renumber an enumeration set although you will invalidate objects where the value is currently being used.

Procedure

1. Right-click anywhere in a field and select **Renumber Enumeration Values**.

The Change Enumeration Value confirmation window explains that you will invalidate objects already using the existing enumeration set.

2. To continue, click **Yes**.
3. In the **Starting Values** field of the Renumber window, click to increase or lower the value.
4. In the **Increment by** field, click to select an increment value.

   If you select a row before renumbering, renumbering starts at the selected row, with this row getting the starting value and each subsequent row getting the next value of increment. Only the selected row and following rows are changed during a renumber action. If renumbering does not start at the topmost row, a possibility of the generated values being identical to the values above the selected row exists. The following error message is displayed:

   ```
   Attempting to change the enumeration values would overlap previous
   enumeration values prior to the selected row.
   ```

Configuration of data references

You can configure an attribute or attribute template to obtain a value from a specified source. This configuration is called a data reference. Attributes that include them are called data reference attributes. You create a data reference for the following sources:

- **PI point**

  You use the PI point value, or the value of summary, or other operations on the point value.

- **PI point array**

  You use a PI point array data reference to produce a single value from an array of PI points.

- **Formula**

  You create formulas to produce a calculation result that defines the attribute value. The calculation itself can include attributes, as well as data reference attributes.

- **Table lookup**

  You define a table internally, import values from an external (non-PI) table, or link to values in an external table.

- **String builder**

  You use substitution parameters and functions to manipulate values and output a string.

- **URI builder**

  You use attribute values and substitution parameters to create dynamic links for attributes. You can use the attribute in a notification or in a client application, such as PI Vision.

Topics in this section

- Attribute configuration strings in PI System Explorer
- Substitution parameters in data references
- PI point data references
- PI point array data references
- Formula data references
- String Builder data references
- Table lookup data references
- URI Builder data references

Attribute configuration strings in PI System Explorer

An attribute configuration string describes a data reference. The syntax of a configuration string depends on the type of data reference.

The configuration string for String Builder data references can contain substitution parameters. For attribute templates, any configuration string can contain substitution parameters. See List of PI AF substitution parameters.

In PI System Explorer, when you select an attribute with a data reference, the configuration string appears in the text box below the **Settings** button. You edit configuration strings directly in the text box.

The following table contains examples of configuration strings for different types of data references:

Type of data reference	Sample configuration string
Formula	`D=Density;V=Volume;[D*V]`
	`A=Attribute3;[A*3];UOM=cm`
PI Point	`\\MyPIDataArchiveServer\sinusoid`
	`\\192.168.0.255\ChocMilkMeter;TimeMethod=TimeRange; RelativeTime=-1h;TimeRangeMethod=Total;ReadOnly=False`
PI Point Array	`\\MyPIDataArchiveServer\Point.1|Point.2|Point.3`
String Builder	`"%Attribute% value is";'Attribute1';`
Table Lookup	`SELECT Density FROM [Material Specifications] WHERE MaterialID = @Product`
URI Builder	`https://MyDataServer.int:443/Vision/#Displays/215915/Mine-Truck-10-Brake-Temp?Asset=\\%System%\%Database%\%Element %&StartTime=03%2F21%2F2016 09:26:00&EndTime=&Mode=kiosk`

Topics in this section

- Configuration strings for PI point data references
- Path syntax for references to other attributes

Configuration strings for PI point data references

The attribute configuration string for PI point data references must contain the path to the point. The string can also contain parameters that specify the value that the data reference returns. If specified in an attribute template, the string can contain parameters that specify the point to create. In the string, you separate the parameters with semi-colons.

Examples

- Simple reference to a point on a PI Data Archive server called `MyPIDataArchiveServer`:
  ```
  \\MyPIDataArchiveServer\sinusoid
  ```
- Configuration string referencing the same point, but with a time retrieval specification and specified units of measure:
  ```
  \\MyPIDataArchiveServer\sinusoid;TimeMethod=ExactTime;UOM=°C
  ```
- Configuration string referencing the same point, but returning a total of point values over a time range:
  ```
  \\MyPIDataArchiveServer\sinusoid;TimeMethod=NotSupported;
  TimeRangeMethod=Total;RateConversion=day
  ```
- Configuration string from an attribute template, using substitution parameters:
  ```
  \\%Server%\%Element%.%Attribute%
  ```
- Same configuration string, but with tag creation enabled and point configuration specified:
  ```
  \\%Server%\%Element%.%Attribute%;ptclassname=classic;pointtype=Float32;
  engunits=m3/s;location1=1;location2=30;location4=1;location5=1;pointsource=R
  ```

Data reference parameters that specify values to return

Configuration strings for PI point data references can include optional parameters that specify the value that the data reference returns. The following table lists available parameters:

Parameter	Syntax	Example
TimeMethod	TimeMethod=*time_method* where *time_method* is one of: • After • AtOrAfter • Before • AtOrBefore • Automatic • ExactTime • Interpolated • NotSupported • TimeRange • TimeRangeOverride	TimeMethod=Automatic
RelativeTime	RelativeTime=[*] +|- integer *time_unit* where *time_unit* is one of: • y • M • d • h • m • s	RelativeTime=-1h
TimeRangeMethod	TimeRangeMethod=*method_name* where *method_name* is one of • Maximum • Minimum • PopulationStandardDeviation • Range • StandardDeviation • StartTime • Total	TimeRangeMethod=Total
TimeRangeMinPercentGood[1]	TimeRangeMinPercentGood= *percentage*	

Parameter	Syntax	Example
`CalculationBasis`[1]	`CalculationBasis=` *calculation_basis* where *calculation_basis* is one of: • `EventWeighted` • `EventWeightedExcludeEarliestEvent` • `EventWeightedExcludeMostRecentEvent` • `EventWeightedIncludeBothEnds` • `TimeWeighted` • `TimeWeightedContinuous` • `TimeWeightedDiscrete`	
`RateConversion`[2]	`RateConversion=`*uom* where *uom* is a defined unit of measure	`RateConversion=minute`
`CaseMethod`	`CaseMethod=`*case_method* where *case_method* is one of: • `StartTime` • `EndTime` • `TimeRange` • `Automatic`	`CaseMethod=EndTime`
`UOM`	`UOM=`*uom* where *uom* is a defined unit of measure	`UOM=°C`
`ReadOnly`	`ReadOnly=`*boolean* where *boolean* is one of: • `true` • `false`	`ReadOnly=false`

[1] Used when specifying **`TimeRangeMethod`**.

[2] Used only when `TimeRangeMethod=Total`.

Data reference parameters that specify PI point to create

In attribute templates that specify new PI points, configuration strings for PI point data references include parameters that specify the type of point to create. Point creation parameters consist of a point attribute and value. To specify a new PI point, attribute templates must include the **`pointtype`** attribute as a parameter; other attributes are optional.

Examples

- Create a PI point of type `Float64` and use default settings for the rest of the point configuration:
  ```
  \\%Server%\%Element%.%Attribute%;pointtype=Float64
  ```
- Create a PI point that sets the point attributes shown in the following table:

Point attribute	Setting
`ptclassname`	`classic`

Point attribute	Setting
pointtype	Float32
pointsource	R
location4	1
location5	2
span	200
zero	1100

```
\\%Server%\%Element%.%Attribute%;ptclassname=classic;pointtype=Float32;
location4=1;location5=2;pointsource=R;span=200;zero=1100
```

Path syntax for references to other attributes

To reference other attributes, you use paths that locate attributes on the server, database, and elements in which they reside.

- Use an absolute path to specify the actual server, database, and element where an attribute is located. For example:
  ```
  \\MyPIDataArchive\MyAFDatabase\MyElement|Attribute1
  ```
- Use a relative path to identify an attribute based on its name and its place in the hierarchy of elements and attributes. For example:
  ```
  ..\Element2|Attribute1
  ```

Components of a path

A path is comprised of parts, with each part representing an object or list of objects. Each part of the path is typically separated as follows: by a single backslash (\), with the following exceptions:

- PI AF attributes and PI AF attribute templates are preceded by the pipe character (|).
- The PI Data Archive is preceded by two backslashes (\\).

You can specify a PI Data Archive and a database by name or by globally unique identifier (GUID). A GUID is a unique 128-bit number produced by Windows to identify a particular component, application, file, or database entry. GUIDs must be specified within curly brackets ({ and }). For example:

```
\\{5c64c379-c182-4f35-8d30-78d8c2f84502}\{5c64c379-c182-4f35-8d30-78d8c2f84503}
```

If you specify both the name and GUID, separate them with a semicolon (;). The first one specified takes precedence in the search, so in the following example, the GUID takes precedence:

```
 \\{5c64c379-c182-4f35-8d30-78d8c2f84502};MySystem\{5c64c379-
c182-4f35-8d30-78d8c2f84503};MyDatabase
```

Topics in this section

- Path syntax for PI Data Archive computers
- Path syntax for PI AF databases
- Syntax for relative paths
- Path syntax for dynamic objects
- Path syntax for collection types

- Path syntax for filters
- Wildcard characters in Name filters

Path syntax for PI Data Archive computers

To indicate a fully qualified path from the PI Data Archive, start the path with two backslashes (\\) followed by the PI Data Archive.

To indicate the current system, start a relative path with two backslashes followed by a period (\\.). For example, to reference a database called `Database2` in the current system, type:

```
\\.\Database2
```

To specify a collection of PI Data Archives, use the following format:

```
\\PIDataArchives[MyPIDataArchiveName]
```

Path syntax for PI AF databases

The PI AF database follows the PI Data Archive specification.

You can start a relative path with a single backslash (\) to indicate the current database. For example, the following paths reference objects in the current database:

```
\Element2
```

```
\Tables[MyTable]
```

Note:
You can access non-database objects external to the PI System. You must identify the collection, as demonstrated by the following example:

```
\\MySystem\Contacts[JSmith]
```

Syntax for relative paths

Use a double period (..) to indicate the parent object. The following example references `Attribute1` of `Element2` that is in the current database:

```
..\Element2|Attribute1
```

The single period (.) represents the current object. You can use it to create a relative path from the current object.

- When the current object is a PI AF attribute, a single period followed by a backslash (.\) represents the owning PI AF element. For example:
  ```
  .\|Attribute1
  ```
- A single period followed by a vertical bar (.|) references a child PI AF attribute, for example:
  ```
  .|Attribute1|Attribute2
  ```
- When the current object is a PI AF element, a relative path is created from the database, for example:
  ```
  \Element1\Element2|Attribute1
  ```

Path syntax for dynamic objects

Paths to dynamic objects are also supported. To create a PI point array attribute with `Sinusoid` and `Sinusoidu` values, for example:

```
PI Point Array.\\piserver2\sinusoid|sinusoidu
```

If a path is to a PI point, a dynamic attribute is created. To create an attribute with the data reference configured to read values from the PI point `Sinusoid`, for example:

```
\\piserver2\sinusoid
```

Path syntax for collection types

Each parent object has a default collection type. For example, a PI Data Archive has a default collection of databases, and a PI AF database has a default collection of elements.

Use a single period enclosed in square brackets ([.]) to represent the default object of the parent object. The following example refers to `Element1` of the default database in the current system:

```
\\.\Databases[.]\Element1
```

Path syntax for filters

A collection filter starts with the at sign (@) followed by the filter name. Supported filters are: `@Category`, `@Description`, `@Index`, `@Name`, `@ReferenceType`, `@Template`, `@Trait`, `@Type`, and `@UOM`.

You must enclose the filter specification in square brackets ([and]).

You can also specify multiple filters; they are evaluated in the order specified. For example:

```
\\MySystem\MyDatabase\Elements[@Template=Tank][@Category=Tutorial]|
Attributes[@Category=Tutorial]
```

Category filter example

The following example returns the `Volume` attribute from all elements in the category, `Tutorial`, that belong to the database called `MyDatabase`:

```
\\MySystem\Databases[MyDatabase]\Elements[@Category=Tutorial]|Volume
```

Type filter example

The following example returns the attributes of `Element1` that are of `Int32` type:

```
\Element1|Attributes[@Type=System.Int32]
```

Unit of measure filter example

The following example returns the attributes of `Element1` that have meters as their unit of measure:

```
\Element1|Attributes[@UOM=meter]
```

Index filter example

Use the index filter `[@Index=`*`int`*`]` or `[`*`int`*`]` to specify the position of the matched object to return (the first item is at index position 1). The index filter must be the last filter specified.

The following example returns the third database in the collection of PI AF databases in the current system:

```
\\Systems[MySystem]\Databases[@Index=3]
```

Beginning with PI AF 2017, you can use a negative number for an index from the end of the collection (for example -1 indicates the last item, -2 indicates the penultimate item).

```
\Element#1\Elements[@Name=Tank*][@Index=-3]
```

The index filter name is optional if another filter is specified before the index filter. For example:

```
\Element#1\Elements[@Name=Tank*][3]
```

Wildcard characters in Name filters

You can place a wildcard asterisk (*) character in the name of any object to match zero or more characters.

> **Note:**
> To match a literal asterisk, use a preceding backslash (*).

Examples

```
\\MySystem\MyDatabase\[@Name=E*]
```

```
\\MySystem\MyDatabase\Elements[@Name=E*][@Index=3]
```

Substitution parameters in data references

Substitution parameters are variables that you place in attribute templates for data references such as PI point, String Builder, and URI Builder. PI AF resolves the substitution parameters when it creates the data reference for an attribute based on that template. For example, you can use substitution parameters:

- To configure a PI point data reference template to use names for tags based on element names built from that template.
- To use the value of a different attribute in the configuration of a PI point property value.

Videos

For information on how to use substitution parameters in templates, watch this video:

https://www.youtube.com/watch?v=8jpOjDRwQf0&rel=0

For information on how to use substitution parameters as you create PI points, watch this video:

https://www.youtube.com/watch?v=pTo9nd_L4vI&rel=0

Topics in this section

- Symbols used in substitution parameters
- Name substitutions
- References to attribute values
- List of PI AF substitution parameters

- Format strings for time substitution parameters
- Find default PI Data Archive servers

Symbols used in substitution parameters

Use the symbols in the following table to build a substitution parameter.

Symbol	Description	Examples
`%...%`	Considers the expression as a substitution parameter.	`%Element%` `%Attribute%`
`.`	Current element or attribute. Use `.\` to navigate down from current element. Use `.|` to navigate to child attributes of the current attribute.	`%.\ChildElement|Attribute%`
`..`	Navigates a level up.	`%..\..\Element%` `%..|Attribute%`
`\`	Separates components of a path, except attributes.	`%..\Element%`
`|`	Separates attributes in a path.	`%..|Attribute%`

`@`	References the value of the object instead of its name.
	Note: Only PI point data references use attribute value substitution syntax. Other data references, such as formula, table lookup, and String Builder have a simpler syntax for referencing attribute values.
		• Attribute value at same level as attribute:
		`%@Attribute%`
		• Attribute value at root level of same element:
		`%@|Attribute%`
		• Attribute value at parent attribute level:
		`%@..|Attribute%`
		• Attribute value at parent element level:
		`%@..\\|Attribute%`
		• Attribute value of child attribute of same element:
		`%@.|Attribute%`
		• Attribute value of child element attribute:
		`%@.\ChildElement\ChildOfChild|Attribute%`
		• Attribute value of primary element of event frame:
		`%@.\Elements[.]|Attribute%`

Name substitutions

When you use a name substitution, PI AF performs a direct substitution of the substitution parameter for whatever that particular parameter represents.

The table in List of PI AF substitution parameters lists the available substitution parameters and what they represent. For example, `%Element%` is a substitution parameter that represents the element name. After you create an element based on that template, you tell PI AF to create the data reference (Attribute-value updates from PI point data references). When PI AF creates

the reference, it substitutes the current element name wherever the configuration says `%Element%`.

Suppose you have a data reference template that references the PI point name:

```
%Element%_TT
```

You create an element named Tank1 that is based on that template. The attribute data reference points to a PI point named:

```
Tank1_TT
```

References to attribute values

When configuring a PI point data reference attribute, you can use substitution parameters to reference the value of another attribute. You can use attribute value references to specify a value in the **Relative Time** field or in the point attribute configuration for automatically-generated PI points. The syntax is:

```
%@AttributeName%
```

where *AttributeName* is the name of the attribute. The @ indicates to substitute the value of the indicated attribute, rather than its name. To reference an attribute that is not a sibling of the current attribute, use the syntax described in Indirect PI point references to define the path to the desired attribute.

PI AF does not update the attribute value over time. When a substitution is used in the tag name, the value is resolved only at the time the data reference is loaded. If a substitution parameter is used to define point attributes, PI AF uses the value of the attribute at the exact time that you created or updated the data reference (see Attribute-value updates from PI point data references for more information). This value is a constant. PI AF does not evaluate that attribute value again, unless you update the data reference.

Note:
The exception to this rule is a value substitution for the **Relative Time**, which is evaluated on every value call to the PI point data reference (see Create ranges of configurable durations).

List of PI AF substitution parameters

PI AF interprets numerous substitution parameters.

Name substitution parameters

The following table lists the name substitution parameters that PI AF supports.

Parameter name	Substitution
%@*path*%	The value of the object's attribute or attribute template, represented by the path.
%Analysis%	Name of analysis, if obtainable from the context.
%Attribute%	Name of the object's attribute or attribute template.
%|Attribute%	Name of the root attribute or attribute template that holds this data reference.
%..|Attribute%	Name of the parent attribute or attribute template that holds this data reference.

Parameter name	Substitution
%Database%	Name of the PI AF database in which the attribute resides.
%Destination%	Name of the destination element for the transfer in which the attribute resides.
%Element%	Name of the element in which the attribute resides. For event frames, this refers to the name of the primary-referenced element.
%\Element%	Name of the root element in which the attribute resides.
%..\Element%	Name of the parent element of the element in which the attribute resides. To retrieve further ancestors, use the `..\` notation, such as `%..\..\Element%`.
%EventFrame%	Name of the event frame in which the attribute resides.
%\EventFrame%	Name of the root event frame in which the attribute resides.
%..\EventFrame%	Name of the parent event frame of the event frame in which the attribute resides. To retrieve further ancestors, use the `..\` notation, such as `%..\..\EventFrame%`.
%Model%	Name of the model, if obtainable from the context.
%Name:*path***%**	Name of the object represented by the path to the object. If the path is not specified, null is returned since the name of the current object is being referenced.
%Server%	Name of the default PI Data Archive for the computer on which you create the attribute. When creating the attribute in PI System Explorer, this is the default PI Data Archive for the computer on which PI System Explorer is running. **Note:** The **%Server%** parameter does not resolve to the computer on which the PI AF database resides. The **%Server%** parameter can resolve to a different PI Data Archive depending on the default in PI AF Client.
%Source%	Name of the source element for the transfer in which the attribute resides.
%System%	Name of the PI AF server or collective where the attribute resides.
%Template%	Name of the template on which the element or event frame is based. For example, if you created element `Valve101` from a template called `Valve`, then the substitute text would be `Valve`.
%\Template%	Name of the root template on which the element or event frame is based.
%..\Template%	Name of the parent template on which the element or event frame is based. To retrieve further ancestors, use the `..\` notation, such as `%..\..\Template%`.
%Transfer%	Name of the transfer in which the attribute resides.

Time substitution parameters

PI AF supports substitution parameters that specify a particular time and time zone, depending on the context. Optionally, you can augment these supported parameters by including a format string that specifies the format of the time string. See Format strings for time substitution parameters.

The following table lists the time substitution parameters that PI AF supports.

Parameter name	Substitution
%Duration%	Time span between the start time and end time, if it can be obtained from the time context. In open event frames, obtains the time span from start time to the current time. The time span uses a different format from other time substitution parameters. • You can use standard formats such as "c" (constant), "g" (general, short), or "G" (general, long), for example: **%Duration:c%**. For more information, see the MSDN article Standard TimeSpan Format Strings (https://msdn.microsoft.com/en-us/library/ee372286). • You can also use a custom time span format, as described in the MSDN article Custom TimeSpan Format Strings (https://msdn.microsoft.com/en-us/library/ee372287). Note that you need to define the symbols that separate days from hours, and so on with a string literal. For example, **%Duration:d\.hh\:mm\:ss%** defines a period (.) as the separator between days and hours, and a colon (:) as the separator between hours, minutes, and seconds.
%EndTime%	Local end time, if obtainable from the time context. For event frame-based objects that do not have an end time yet specified, the result is an empty string.
%StartTime%	Local start time, if obtainable from the time context.
%Time%	Local time, if obtainable from the time context.
%UtcEndTime%	Coordinated universal (UTC) end time, if it can be obtained from the time context. For event frame-based objects that do not have an end time yet specified, the result is an empty string.
%UtcStartTime%	Coordinated universal (UTC) start time, if it can be obtained from the time context.
%UtcTime%	Coordinated universal (UTC) time, if it can be obtained from the time context.

ID substitution parameters

The following table lists the ID substitution parameters that PI AF supports.

Parameter name	Substitution
%AnalysisId%	ID of the analysis with which the attribute is associated.
%AttributeId%	ID of the attribute that holds this data reference.
%|AttributeId%	ID of the root attribute or root attribute template in which the attribute resides.
%..|AttributeId%	ID of the parent attribute or parent attribute template in which the attribute resides. Further ancestors can be retrieved by using the `..|` notation, such as `%..|..|AttributeId%`.
%DatabaseId%	ID of the database in which the attribute resides.
%ElementId%	ID of the element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element.
%\ElementId%	ID of the root element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element of the root event frame in which the attribute resides.

Parameter name	Substitution
%..\ElementId%	ID of the parent element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementId%`.
%EventFrameId%	ID of the event frame in which the attribute resides.
%\EventFrameId%	ID of the root event frame in which the attribute resides.
%..\EventFrameId%	ID of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFrameId%`.
%Id:*path***%**	ID of the object represented by the path to the object. If the path is not specified, the ID of the current object is used. .
%ModelId%	ID of the model in which the attribute resides.
%SystemId%	ID of the PI AF server in which the attribute resides.
%TransferId%	ID of the transfer in which the attribute resides.

Description substitution parameters

The following table lists the description substitution parameters that PI AF supports.

Parameter name	Substitution
%Description:*path***%**	Description of the attribute represented by the path to the attribute. If the path is not specified, the description of the current attribute is used.
%|Description%	Description of the root attribute or root attribute template in which the attribute resides.
%..|Description%	Description of the parent attribute or parent attribute template in which the attribute resides. Further ancestors can be retrieved by using the `..|` notation, such as `%..|..|Description%`.
%ElementDescription%	Description of the element in which the attribute resides. For event frames, this refers to the description of the primary referenced element of the event frame in which the attribute resides.
%\ElementDescription%	Description of the root element where the attribute resides. For event frames, this refers to the description of the primary referenced element of the root event frame in which the attribute resides.
%..\ElementDescription%	Description of the parent element in which the attribute resides. For event frames, this refers to the description of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementDescription%`.
%EventFrameDescription%	Description of the event frame in which the attribute resides.
%\EventFrameDescription%	Description of the root event frame in which the attribute resides.
%..\EventFrameDescription%	Description of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFrameDescription%`.

Path substitution parameters

Path substitutions cannot generally be used to create references to other attributes, because they contain backslash characters. The Path substitution parameters are most useful with String Builder data references when you construct paths for output into strings. The path that is produced does not include the PI AF server or database.

The following table lists the path substitution parameters that PI AF supports.

Parameter name	Substitution
`%ElementPath%`	Path of the base element, the element of an attribute, or the primary referenced element of an event frame.
`%..\ElementPath%`	Path of the parent element in which the attribute resides. For event frames, this refers to the path of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementPath%`.
`%EventFramePath%`	Path of the event frame, or the element of an attribute of an event frame.
`%..\EventFramePath%`	Path of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFramePath%`.

Environment variable parameter

The following table lists the environment variable parameter that PI AF supports.

Parameter name	Substitution
%*Environment Variable***%**	Value of the matching system-environment variable. For example, **`%COMPUTERNAME%`** is replaced with the name of the computer on which the data reference is executing.

Format strings for time substitution parameters

In time substitution parameters, you can include a format string that specifies the format of the time string.

Standard date and time format

Use any standard format string supported by the `DateTime.ToString` method, described in the MSDN article Standard Date and Time Format Strings (https://msdn.microsoft.com/en-us/library/az4se3k1%28v=vs.110%29.aspx#Sortable). In the specification of the time substitution parameter, separate the substitution and the format string with a colon. For example, `%TIME:d%` specifies the local time in a short-date pattern.

The following table provides an abbreviated description of `DateTime` format strings:

Format specifier	Description	Example
d	Short date pattern	6/15/2009 (en-US) 15/06/2009 (fr-FR) 2009/06/15 (ja-JP)

Format specifier	Description	Example
D	Long date pattern	Monday, June 15, 2009 (en-US) Montag, 15. Juni 2009 (de-DE)
f	Full date/time pattern (short time)	Monday, June 15, 2009 1:45 PM (en-US)
F	Full date/time pattern (long time)	Monday, June 15, 2009 1:45:30 PM (en-US)
g	General date/time pattern (short time)	6/15/2009 1:45 PM (en-US) 15/06/2009 13:45 (es-ES) 2009/6/15 13:45 (zh-CN)
G	General date/time pattern (long time)	6/15/2009 1:45:30 PM (en-US) 15/06/2009 13:45:30 (es-ES) 2009/6/15 13:45:30 (zh-CN)
M, m	Month/day pattern	June 15 (en-US)
O, o	Round-trip date/time pattern	2009-06-15T13:45:30.0000000-07:00
R, r	RFC1123 pattern	Mon, 15 Jun 2009 20:45:30 GMT
s	Sortable date/time pattern	2009-06-15T13:45:30
t	Short time pattern	1:45 PM (en-US)
T	Long time pattern	1:45:30 PM (en-US)
u	Universal sortable date/time pattern	2009-06-15 20:45:30Z
U	Universal full date/time pattern	Monday, June 15, 2009 8:45:30 PM (en-US)
Y, y	Year month pattern	June, 2009 (en-US)
Any other single character	Unknown specifier	

Custom date and time format

You can also construct custom patterns in the date and time format string, using time specifiers described in the MSDN article Custom Date and Time Format Strings (https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx). For example, `%TIME:yyyy/MM/dd HH:mm:ss.ffffff%` produces output similar to 2015/03/31 09:28:03.843512.

Find default PI Data Archive servers

The `%Server%` substitution parameter resolves to the name of the default PI Data Archive server for the computer on which you create the attribute. If you are creating the attribute in PI System Explorer, this is the default PI Data Archive server for the computer on which PI System Explorer is running.

Note:
Because the `%Server%` parameter can resolve to a different PI Data Archive server based on the client, it is not recommended for installations that may have more than one PI Data Archive server available.

Procedure

1. On the computer where PI System Explorer is running, open the PI SDK Utility. With PI Data Archive installation defaults, you would choose **Start All Programs** > **PI System** > **AboutPI-SDK**
2. In the PI SDK Utility window, choose **File** > **Connections**.
3. In the **PI Connection Manager** window, choose **Connections** > **Options**. The Connection Options window opens with the default server listed under **Default Server Settings**.

PI point data references

A PI point data reference is a reference from a PI AF attribute to a PI point. The value of the data reference attribute can be the same as the point value, or it can be the result of a calculation based on point values. You can create a PI point data reference for:

- An element attribute or an element attribute template
- A transfer attribute or a transfer attribute template
- An event frame attribute or an event frame attribute template

A PI point data reference for an attribute template has additional features. When you create a PI point data reference in an attribute template, you can:

- Automatically create tags referenced by attributes based on the template.
- Create a naming scheme for attributes based on the template so that attributes built from it have unique names that conform to a consistent structure.

To reference the PI point, you use a direct or an indirect reference. There are two quick methods to create a direct reference, whereas you need to use the PI Point Data Reference window to create a more complex direct reference or an indirect reference.

Videos

For information on how to set up PI point data references, watch this video:

https://www.youtube.com/watch?v=CRejcRe22SA&rel=0

For information on UOMs in PI point data references, watch this video:

https://www.youtube.com/watch?v=023T5si2Dtw&rel=0

Topics in this section

- Direct PI point references
- Indirect PI point references
- Create direct PI point data references from Tag Search results
- Configure direct or indirect PI point data references
- Unit of measure considerations
- Configuration of retrieval methods for attribute values
- PI point value updates from a data reference
- Attribute-value updates from PI point data references

- Configure automatic creation of PI points
- Edit PI point properties
- Preview substitution parameters in PI point data references

Direct PI point references

In the configuration string, a direct point reference uses two backslashes (\\) to reference a PI point on a PI Data Archive. For example, the following configuration string references the *sinusoid* point on a PI Data Archive called MyPIServer:

```
\\MyPIServer\sinusoid
```

For attribute templates, you can also use substitution parameters in the reference. For example, the following configuration string references the *sinusoid* point on the default PI Data Archive of the PI AF database for the attribute:

```
\\%Server%\sinusoid
```

For more information on substitution parameters, see Substitution parameters in data references.

Indirect PI point references

You can configure a PI point data reference to point at another attribute. The referenced attribute must itself be a PI point data reference. This is called an *indirect reference* to whatever PI point the target attribute references. It makes sense to reference by attribute when you need multiple attributes that each use data from the same PI point.

For example, suppose you have an attribute called Level that references a PI point registering a tank level. You want three child attributes to hold the daily average, minimum, and maximum tank levels. If you configure the three child attributes to reference the Level attribute, you can later change the Level attribute to reference a different point and you do not have to reconfigure child attributes. Used in combination with templates, this significantly reduces the amount of configuration required per element instance.

Relative path

To reference another attribute, the configuration string uses a *relative path*. The relative path identifies a data reference attribute based on its name and its place in the hierarchy of elements and attributes. For the PI point data reference, the path must include an attribute path designation (| or ..) in the configuration string so that it is distinguished from a PI point reference.

The following table shows typical configurations for indirect references:

Object	Sample syntax
Top level attribute of same element	`|topLevelAttribute`
Parent attribute	`..`
Child attribute	`.|childAttribute`
Sibling attribute (when not a top level attribute)	`..|siblingAttribute`
From event frame to primary element	`.\Elements[.]|Attribute`
From event frame to attribute of parent element	`.\Elements[.]\..\\|Attribute`
From event frame to attribute of child element	`.\Elements[.]\ChildElementName|Attribute`
Database path	`\TopLevelElement|myAttribute`
Full path	`\\myServer\myDatabase\myElement|myAttribute`

For further details about relative path syntax, see Path syntax for references to other attributes.

Create direct PI point data references from Tag Search results

You can drag a PI point from Tag Search results in the palette to set the name, description (if the PI point has a descriptor), and type of a PI point data reference. If the engineering units property of the PI point matches an existing unit of measure, including case, the **Default UOM** field is also set.

Procedure

1. To configure a PI point data reference:
   - In the Elements browser, select the desired element.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. Select **View** > **Palette** > **Tag Search**, or press CTRL+SHIFT+8.
4. In the Tag Search window, create a search query to retrieve the PI points you want to use as data references. For more information on searching for PI points, see Search for PI points and Syntax for PI point searches.
5. Click **Search** to retrieve the points that match into the results table.
6. Do one of the following:
   - To create a new attribute, drag a PI point from search results onto the **Attributes** grid. A new attribute configured as a data reference based on the PI point is added to the grid.
   - To configure an existing attribute, select the attribute in the viewer and drag a PI point from search results to the attribute **Settings** field to configure the attribute as a data reference based on the PI point.
7. Click **Check In** to save the data reference.

Configure direct or indirect PI point data references

Use the PI Point Data Reference window to configure PI point data references on element attributes.

Procedure

1. From the **Attributes** tab in the viewer, select an attribute.
2. In the attribute configuration section, select **PI Point** from the **Data Reference** list.
3. Click **Settings**.

   Tip:
   You can click in the **Settings** field and enter a tag name to create a direct PI point data reference in `\\serverName\tagname` format. PI System Explorer retrieves the tag name from the default PI Data Archive. For example, if you enter `sinusoid`, the PI point data reference displays `\\yourDefaultDataArchive\sinusoid`.

4. In the PI Point Data Reference window, choose from the following actions to create the data reference.

To ...	Do this ...
Create a direct PI point data reference	a. Leave the default server, or choose a different PI Data Archive from the **Data server** list. **Note:** If the desired PI Data Archive does not appear in the list, click ... and select a PI Data Archive from the PI Data Archives window. b. In the **Tag name** field, type the point name, or click to search for a point on the selected PI Data Archive. **Tip:** You can enter a substitution parameter instead of the tag name. If you are creating a reference in an attribute template, you can use substitution parameters in both the **Data server** and **Tag name** fields. See List of PI AF substitution parameters.
Create an indirect PI point data reference	a. Choose the **Attribute** option. b. Type a relative path to the attribute, or choose one from the drop-down list of attributes that have PI Point data references. To type in a path, use the syntax described in Indirect PI point references. The attribute that is referenced must also have a PI point data reference.

5. Optional. Specify the units that the referenced PI point uses. See Unit of measure considerations.

> **Note:**
> For time weighted totals on referenced PI Point data reference attributes, ensure that you select a UOM from the **Source Units** list that matches the UOM of the source attribute. Even if the default source units appear to match the UOM of the source attribute, do not select **<Default>** but instead select the appropriate hardcoded UOM from the list.

6. In the **Value retrieval methods** section, specify how the attribute gets its value. For example, the attribute value could be the same as the point value, or it could be a result of a calculation on the point value. See Configuration of retrieval methods for attribute values.
7. Use the **Read Only** check box to specify whether you want PI AF to write the attribute value back to the source point. By default, PI AF does not write data to the referenced PI point (the **Read Only** check box is checked). See PI point value updates from a data reference.

Unit of measure considerations

When you define a PI point data reference you can optionally specify the units that the referenced PI point uses. If possible, PI AF automatically changes the attribute type to a value type that is compatible with the specified units.

However, if the attribute is defined by an attribute template, PI AF cannot change the type. Instead, PI AF attempts to convert the value to the template's value type. If PI AF cannot convert the value, it generates an error.

Configuration of retrieval methods for attribute values

Client applications request attribute values for a specific time or for a time range. For example, in PI ProcessBook, the display can optionally provide a time range context (a time range symbol, such as a trend, must be present on the display to enable reception of a time range). You typically configure the data reference to expect either a time or a time range. The attribute value will then be either:

- The value of the point at a specific time. See Configure value retrieval by time.
- The result of a calculation on the point's values over a time range. See Configure value retrieval by time range. For example, the attribute value could be the average of the point values over an hour.

Videos

For information on how to retrieve aggregate values in PI point data references, watch this video:

https://www.youtube.com/watch?v=GU3kp7W7a6w&rel=0

For information on how to retrieve a single value in a PI point data reference, offset by a time period, watch this video:

https://www.youtube.com/watch?v=4Mv5aZWz2j8&rel=0

Topics in this section

- Relative time
- Configure value retrieval by time

- Configure value retrieval by time range
- Time ranges for event frames and transfers

Relative time

Relative time expressions are some number of a number of days (`d`), hours (`h`), minutes (`m`), or seconds (`s`), specified with either a leading plus sign (+) or a leading minus sign (-). The default starting point for relative time is the current time. Therefore, a time of **-8h** is eight hours before the current time. Fractional times are supported. For example, use `-1.5d` for one and one-half days. These are all valid relative times:

```
+1d
-24h
-3.25m
+24s
```

Relative time expressions can contain only one operator, either + or -. For example, this is *not* supported:

```
-1d+1h
```

Configure value retrieval by time

A client application request for an attribute value includes a time. Specify how to interpret that time to retrieve values for the attribute.

Procedure

1. Select an option from the **By Time** list.

Option	Description
After	Returns the first recorded value after the time requested by the client application.
At or Before	Returns a recorded value at the time requested by the client application. If no value exists at the specified time, returns the previous recorded value.
At or After	Returns a recorded value at the time requested by the client application. If no value exists at the specified time, returns the next recorded value.
Automatic	A continuous point (step attribute = 0) is treated as Interpolated, whereas a discrete point (step attribute = 1) is treated as "At or Before."
Before	Returns the first recorded value before the time requested by the client application.
Exact Time	Returns a recorded value at the time requested by the client application. If no recorded value exists at that time, an error is returned.
Interpolated	Returns an interpolated value for the time requested by the client application. Discrete points (step attribute = 1) carry the previous value forward.

> **Note:**
> Do not select the **Not Supported**, **Time Range**, and **Time Range Override** options. These options are for attribute values based on time range calculations.

2. Leave the **Relative Time** field empty, unless you want to set up a time delay.
3. From the **By Time Range** list, select **End Time**.

   If the client application sends a time range rather than a specific time, PI AF returns an error message as the attribute value.
4. Click **OK**.

Set up a time delay

If an attribute is configured to return the value at a specific time, you can configure a time delay from the time requested. This can be useful when you are creating dead-time delayed attributes.

Procedure

1. Configure the **Value Retrieval** settings, as described in Configure value retrieval by time.
2. In the **Relative Time** field, type the relative time of the delay. Use a PI relative time expression. See Relative time.

   > **Note:**
   > You can use substitution parameters to read the relative time from the value of another attribute. See References to attribute values.
3. Click **OK**.

Configure value retrieval by time range

If you want the attribute value to be the result of a summary calculation over a time range, configure the value retrieval by time range.

Procedure

1. Select one of the following options from the **By Time** list.

Option	Description
Not Supported	If the client application sends a time instead of a time range, PI AF returns an error message as the attribute value.
Time Range	Create a default time range to use if the client application sends a time instead of a time range. If you choose this option, you must type a PI relative time expression in the **Relative Time** field. See Create default time ranges for element attributes for details.
Time Range Override	Specify a time range that always overrides the time range specified by the client application. You can specify a fixed duration for the range (Create ranges of configurable durations) or you can calculate the duration dynamically (Configure dynamic time range calculations).

2. Select one of the following options from the **By Time Range** list.

Option	Description
Average	Returns the average value over the time range.
Count	Returns the event count over the time range, when **Calculation Basis** is set to **Event Weighted**. Returns the sum of event time duration over the time range, when **Calculation Basis** is set to any of the time weighted options.
Delta	Returns the difference in value from the end of the time range to the start of the time range.
End Time	Returns the value at the end of the time range.
Maximum	Returns the maximum value over the time range. **Note:** The timestamp value displays the time that the maximum value occurred.
Minimum	Returns the minimum value over the time range. **Note:** The timestamp value displays the time that the minimum value occurred.
Population Standard Deviation	Returns the population standard deviation over the time range.
Range	Returns the range of values over the time range (Maximum-Minimum)
Standard Deviation	Returns the standard deviation over the time range.
Start Time	Returns the value at the Start Time of the time range. **Note:** If you choose the **Start Time** option, you cannot choose **Not Supported** in the **By Time** list, otherwise the following error message is displayed: `The TimeMethod must be supported to support the TimeRangeMethod of "Start Time".`
Total	Returns a totalization over the time range.

3. If the **Calculation Basis** list is enabled, select one of the following options.

Note:
If you select one of the time weighted options for a totalization, source units convert to a rate. Select a unit of time from the list next to the **Source Units** field.

Option	Description
Event Weighted	Evaluates values with equal weighting for each event. No interpolation is done. At least one event must be within the time range to perform a successful calculation. Two events are required for standard deviation. In handling events at the boundary of the calculation, PI AF uses the following rules: ◦ Events at both boundaries are used when there is only one calculation interval. ◦ Events at start time are included when multiple intervals exist and the intervals are in ascending time order. ◦ Events at the end time are included when multiple intervals exist and the intervals are in descending time order.
Event Weighted Exclude Most Recent Event	Behaves the same as **Event Weighted**, except in the handling of events at the boundary of summary intervals in a multiple intervals calculation. Use this option to prevent events at the intervals boundary from being double count at both intervals. With this option, events at the end time (most recent time) of an interval are not used in that interval.
Event Weighted Exclude Earliest Event	Similar to the **Event Weighted Exclude Most Recent Event** option. Events at the start time (earliest time) of an interval are not used in that interval.
Event Weighted Include Both Ends	Events at both ends of the interval boundaries are included in the event-weighted calculation.
Time Weighted	Weights the values in the calculation by the time over which they apply. Interpolation is based on the **`step`** attribute of the point. Interpolated events are generated at the boundaries if necessary.
Time Weighted Continuous	Applies weighting as in **Time Weighted**, but does all interpolation between values as if they represent continuous data (standard interpolation), regardless of the **`step`** attribute of the point.
Time Weighted Discrete	Applies weighting as in **Time Weighted** but does interpolation between values as if they represent discrete, unrelated values (stair-step plot), regardless of the **`step`** attribute of the point.

4. In the **Min percent good** field, enter a value between 0 and 100. This is a percentage representing the minimal amount of time a value must be good before a summary value is returned. For time weighted calculations, the percentage is based on time. For event weighted calculations, the percent is based on event count.
5. Click **OK**.

Topics in this section

- Create ranges of configurable durations
- Configure dynamic time range calculations
- Create default time ranges for element attributes

Create ranges of configurable durations

To guarantee that time ranges are always of the same length, configure time range overrides.

Procedure

1. From the **By Time** list, select **Time Range Override**.
2. In the **Relative Time** field, type a PI **relative time** expression (Relative time).

   PI AF creates a time range based on the end time supplied by the client application and the offset defined by the relative time.

   > **Note:**
   > You can use substitution parameters to read the relative time from the value of another attribute (References to attribute values). If you do this, PI AF evaluates the referenced attribute value at the time of each data request.

Example

To create a one-hour rolling average, you would use the following settings:

- From the **By Time** list, select **Time Range Override**.
- In the **Relative Time** box, type `-1h`.
- From the **By Time Range** list, select **Average**.

With this configuration, the data reference always computes a 1-hour average, even when the client application specifies a different time range.

Configure dynamic time range calculations

Use a PI AF attribute to calculate the time range based on variable factors. You can use the attribute value to determine the time range dynamically. The attribute values must be valid PI relative times, as described in Relative time.

Procedure

1. Configure an attribute to generate the PI relative times. The attribute might reference an enumeration set containing PI relative times, or it might construct PI relative times based on a calculation.
2. Configure the PI point data reference.
   a. From the **By Time** list, select **Time Range Override**.
   b. In the **Relative Time** field, type: `%@AttributeName%` where **AttributeName** is the name of the attribute that generates the PI relative time values.
   c. From the **By Time Range** list, select a calculation method.

Results

With this configuration, the data reference uses the calculated relative time as a time offset that determines the time range.

See Example variable end time to step through an example.

Example variable end time

This example creates a data reference with a variable end time.

Procedure

1. Create an enumeration set, `MyEnumSet`, that contains these three values:

```
-30m
-1h
-2h
```

> **Note:**
> Values must be valid relative time expressions, as described in Relative time.

2. Create an attribute, called `TimeRangeAttribute`, that references the `MyEnumSet` enumeration set.

   You specify the enumeration set when configuring the attribute. From the **Value Type** list, select **Enumeration Sets** > **MyEnumSet**.

3. Create another attribute, called `DataReferenceAttribute` with a configured PI point data reference:

   a. For the PI point, choose any point that has a numeric value. In this example, we will use the `sinusoid` point on a PI Data Archive named PISRV1.

   b. From the **By Time** list, select **Time Range Override**.

   c. In the **Relative Time** box, type: `%@TimeRangeAttribute%`

   d. From the **By Time Range** list, select **Average**.

Results

The `TimeRangeAttribute` attribute lets you choose the time interval for the calculation.

Create default time ranges for element attributes

You can create a default time range for element attributes. A default time range is a time range that PI AF uses when a client application provides a specific time, rather than a time range.

> **Note:**
> Event frame and transfer attributes treat time ranges differently, as described in Time ranges for event frames and transfers.

Procedure

1. In the PI Point Data Reference (or PI Point Array Data Reference) window, use the following value retrieval settings:
   a. From the **By Time** list, select **Time Range**.
   b. In the **Relative Time** field, type a PI relative time expression, as described in Relative time.

      > **Note:**
      > You can also use substitution parameters to read the relative time from the value of another attribute, as described in References to attribute values.

   PI AF creates a time range based on the time specified by the client application and the offset defined by the relative time.

Example

A totalization meter requires a time range in order to deliver a totalized value. If the client application sends a specific time, you want to create a one-hour time range that ends in the specified time. To do that, you would use these settings:

1. From the **By Time** list, select **Time Range**.
2. In the **Relative Time** field, type:

   ```
   -1h
   ```

3. From the **By Time Range** list, select **Total**.

Time ranges for event frames and transfers

Event frames and transfers, by their nature, have a time range associated with them. When a calling application requests a value from an attribute, the value returned depends on the data reference and its configuration:

- When the calling application supplies no time context, PI AF calculates the attribute value using the time range of the event frame or transfer. For example, if an attribute contains a PI point configured with a time range summary option, the summary will be performed over the time range of the event frame or transfer.

If an attribute contains a PI point configured with only the default time option, PI AF returns the value of the point at the end time of the event frame.

- When the calling application supplies a single timestamp, PI AF returns the value at the specified time, even if the time is outside the time range of the event frame or transfer.
- If the calling application supplies a time range, PI AF uses the intersection of the supplied time range and the time range of the event frame. If there is no intersection, a "No Data" event is returned.

  The shaded area in the following figure represents the intersection between an event frame time range and a requested time range. For attributes that contain a PI point configured with a time range summary option, the summary is performed over this intersection. If the attribute contains a PI point configured with only the default time option, the value at the start time of the intersection is returned.

Intersection between event frame time range and requested time range

PI point value updates from a data reference

An attribute with a PI point data reference can write a value to a PI point. In the PI Point Data Reference configuration window, the **Read only** check box determines whether the attribute can write data to the referenced PI point. By default, the check box is selected (Read only = True), meaning that the PI AF attribute cannot change the value of the PI point.

In rare cases, you might want to write the attribute value back to the point value. For example, if the attribute value is the result of a calculation or analysis, you can update the PI point with the result of that analysis. In this case, you would clear the **Read only** check box to allow the attribute to write its value to the point.

Note:
Applications that use the AF SDK **`UpdateValue`** method, such as the PI Analysis Service, can write data to the PI point regardless of whether the **Read only** check box is selected or not.

Attribute-value updates from PI point data references

When you create an attribute based on a template with a data reference, PI AF does not automatically create the PI point data references associated with that element. To create the data references, you must right-click the element in the Elements (or Event Frames) browser and click **Create or Update Data Reference**. PI AF replaces all substitution parameters with the corresponding values at that moment, and creates and locks in the data reference.

Data reference lock-in

When a data reference is locked in, changes to the attribute template no longer affect instances of the attribute. Substitution parameters are also no longer evaluated. However, the following fields are exceptions:

- An attribute value reference is entered in the **Relative Time** field, as noted in References to attribute values.
- Any point attribute change made in the Tag Creation Settings window.

For example, if a point attribute change is made or a point attribute is defined by substitution parameters, the data reference of derived attributes will still be affected whenever you right-click an element and click **Create or Update Data Reference**.

Data reference reset

After an attribute value has been locked in, you can use the **Reset to Template** feature to reset an element to its template's original properties, as described in Attribute-value reset to original properties.

Attribute indicators for updates of PI point data references

PI System Explorer employs distinctive visual indicators for attributes that support the **Create or Update Data Reference** option. A visual indicator is also displayed when an attribute no longer appears to be configured correctly, such as after a server or a tag name has been changed.

The following visual indicators for the **Create or Update Data Reference** option are available:

Icon	Description
	The attribute is associated with a template that contains point creation rules. Use the **Create or Update PI Point** option to create the tag configuration based on the template. Additionally, the point name and ID is resolved.
	The attribute's data reference configuration differs from what is stored in the PI AF database. This can occur when a point is renamed or when the underlying PI Data Archive is replaced. Use the **Create or Update PI Point** option to update the configuration.
	The attribute is a configuration item. See Configuration Item attribute property for more information. **Note:** The **Create or Update Data Reference** option is not applicable for configuration item attributes.

When you review element attributes on the **Attributes** tab, the first entry in the column at left of a data reference attribute's name indicates whether you can run the **Create or Update Data Reference** option. You initiate the **Create or Update Data Reference** option by right-clicking an element or an attribute and clicking **Create or Update PI Point**.

The **Create or Update Data Reference** operation updates a PI point data reference with any changes that have occurred to the PI point; substitution parameters are resolved, and server and point IDs are updated. If the PI point does not yet exist, and the configuration contains a point creation option, it is created.

Attribute update example

When you see the indicator displayed for an attribute, you can update its configuration by right-clicking and clicking **Create or Update PI Point**.

> **Note:**
> The icon does not determine beforehand if any changes to the tag or its configuration will occur.

Attribute indicator with tooltip

As the tooltip shown above indicates, when you update the configuration, point name substitution rules are resolved and the point ID is stored into the attribute's metadata. This results in faster subsequent initialization.

Configure automatic creation of PI points

In an attribute template, you can configure a PI point data reference to create PI points automatically.

Procedure

1. In the Library browser, select an element template.
2. Click the **Attribute Templates** tab.
3. Add a new attribute template to the element template, as described in Create attribute templates.
4. From the **Data Reference** list, select **PI Point**.
5. Click **Settings**.

6. In the **Tag name** field in the PI Point Data Reference window, set up the PI point names for the generated points. Use substitution parameters to enable PI AF to build point names automatically for the points it generates.
   a. Click ▸ to select name substitution and related attribute values from a context menu. For more information on name substitution values, see List of PI AF substitution parameters.

      Note:
      If the attribute value substitution is a string that contains spaces, you must surround the entire tag name in double quotes. Concatenate the entire string first, then substitute the value into the tag name portion of the string with String Builder. For example:

      ```
      "\\%Server%\%Element%.%ID%.%@ConcatStringName%;pointtype=String"
      ```

   b. Select the **Tag Creation** check box.
   c. Click ···.
   d. In the Tag Creation Settings window, specify a point class and point type in the **Point Class** and **Point Type** fields. For more information on PI point classes and attributes, see "PI point classes and attributes" in Live Library (https://livelibrary.osisoft.com).
   e. Default attribute values are displayed for the selected point class. You can edit the default PI point attributes, or you can click **Import** to import PI point attribute values from an existing point that you locate in the Tag Search window.
7. To preview how the selected substitution parameters resolve, click the **Select example instance** link in the **Preview** section.
8. In the Find Derived Elements for *Element_Template* window, select a sample element.
9. Click **OK**.

   The **ConfigString** and **Value** fields display the resolution of the substitution parameters you selected, as well as the tag creation settings.
10. To save your work, press CTRL-S or click **Check In**.
11. Create a new element based on your element template and click on the **Attributes** tab.
12. Right-click the PI point data reference attribute and click **Create or Update Data Reference**.
13. To save your work, press CTRL-S or click **Check In**.

Results

All the points that the new elements reference are created. This only works because the elements were based on a template and the attribute template for the data reference uses the Tag Creation option.

After you finish

After the PI points are created, it might take some time before values are written into PI Data Archive by the specified interface. Until then, the values are displayed as **Pt. Created**.

Edit PI point properties

Edit a PI point property when you need to provide more information on a PI point than default tag property values. For example, it can be helpful to provide a descriptor for a PI point that is created as output from an analysis.

Procedure

1. In the Elements browser, locate the element that contains the PI point you want to edit.
2. Click the **Attributes** tab and select the attribute with the **PI Point** data reference.
3. Click **Settings**.
4. In the PI Point Data Reference window, click .
5. In the PI Point Properties window, choose from the following actions:

To ...	Do this ...
Edit a property value	a. Click the **Value** field beside a property. b. Enter a revised value.
Restore an edited value to PI Data Archive default	a. Click the **Value** field beside an edited property. b. Click and select **Restore default value**.

6. Click **OK**.

Preview substitution parameters in PI point data references

Preview how substitution parameters resolve as you configure a PI point data reference in an attribute template.

Procedure

1. In the Library browser, select an element template.
2. Click the **Attribute Templates** tab.
3. Add a new attribute template to the element template, as described in Create attribute templates.
4. From the **Data Reference** list, select **PI Point**.
5. Click **Settings**.

   Default substitution values are displayed in the **Tag name** field.
6. Use substitution parameters to enable PI AF to build point names automatically for the points it generates. Click to select name substitution and related attribute values from a context menu. For more information on name substitution values, see List of PI AF substitution parameters.
7. To preview how the selected substitution parameters resolve, click the **Select example instance** link in the **Preview** section.
8. In the Find Derived Elements for *Element_Template* window, select a sample element.
9. Click **OK**.

The **ConfigString** and **Value** fields display the resolution of the substitution parameters you selected, as well as any tag creation settings and value retrieval methods (if selected).

10. Choose one of the following actions:
    - To confirm the PI point data reference configuration, click **OK**.
    - To change configuration settings, click **Cancel** and start over by clicking **Settings**.

PI point array data references

You use the PI point array data reference to link attributes of elements to arrays of PI points for either reading or writing their values. Each tag name specified corresponds to a single entry in the array. Configure the PI point array data reference as you would the PI point data reference, with the exception of the **Tag Names** field in the PI Point Array Data Reference window.

Video

For information on how to set up PI point array data references, watch this video:

https://www.youtube.com/watch?v=ozyS9QFFZgM&rel=0

Create PI point array data references

Create a PI point array data reference to link attributes of elements to arrays of PI points for either reading or writing their values.

Procedure

1. To configure a PI point array data reference:
    - In the Elements browser, select the desired element.
    - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. In the attribute configuration panel, choose **PI Point Array** from the **Data Reference** list.
4. Click the **Settings** button.
5. In the PI Point Data Array Reference window, leave the default server, or choose a different PI Data Archive from the **Data server** list.

    Note:
    If the desired PI Data Archive does not appear in the list, click ... and select a PI Data Archive from the PI Data Archives window.

6. In the **Tag name** field, choose from the following actions:
    - To enter PI points directly, enter point names separated by the pipe symbol (|). For example:

      ```
      CDM158|CDT158|SINUSOID
      ```

    - To search for PI points, click and enter search criteria in the Tag Search window. Select multiple PI points on the selected PI Data Archive.
7. Optional. Specify the units that the referenced PI points use. See Unit of measure considerations.

8. In the **Value retrieval methods** section, specify how the attribute gets its value. For example, the attribute value could be the same as the point values, or it could be a result of a calculation on the point values. See Configuration of retrieval methods for attribute values.
9. Use the **Read Only** check box to specify whether you want PI AF to write the attribute value back to the source points. By default, PI AF does not write data to the referenced PI points (the **Read Only** check box is checked). See PI point value updates from a data reference.

Formula data references

You use formula data references to create custom calculations based on other element attributes. Calculations can be a single formula or a sequence of calculations. Formula data references can have multiple input attributes.

Videos

For information on how to create formula data references, watch this video:

https://www.youtube.com/watch?v=vReXkkMueIo&rel=0

For information on UOMs in formula data references, watch this video:

https://www.youtube.com/watch?v=0A_Uy5_nbCs&rel=0

Topics in this section

- Formula data reference operators
- Formula data reference functions
- Units of measure in formula data references
- Configure formula data references

Formula data reference operators

You can use the following operators in formula data references (in order of precedence):

Operator	Precedence
Parenthesis	9 (calculated first)
Unary Minus	8
^	7
* / mod	6
+ -	5
< > <= >= == <>	4
NOT	3
AND	2
OR	1 (calculated last)

You cannot use the assignment operator = at the beginning of any formula.

Note:
Formula data reference syntax uses == (two equals signs) to indicate equality and = (single equals sign) for assignment. Analytics (and performance equation) syntax uses = to indicate equality and := for assignment.

Compound operators

You can choose **If-Then-Else**, a compound operator with operands used as follows:

```
If expr0 Then expr1 Else expr2
```

where *expr0*, *expr1*, and *expr2* are expressions. If *expr0* is true, the value of *expr1* is returned; otherwise, the value of *expr2* is returned.

Formula example

This sample formula calculates the daily average cost of reagent per tonne processed, as shown in the **Value** field for **Grinding Media** in the following mineral processing illustration:

```
A=.|Addition (Daily Total);B=.|Unit Cost;C=Tonnage (Past 24 Hours);[if C ==0
then digstate("No Data") else A*B/C]
```

Sample formula data reference

Formula data reference functions

PI AF supports the following functions in a formula data reference.

Function	Description
abs(X)	Absolute value of X.
acos(X)	Arc cosine of X.
asin(X)	Arc sine of X.

Function	Description
atan(X)	Arc tangent of X.
badval(x)	Returns 1 if the value has a bad status, has a floating point value of not a number (NaN), or is not a numeric value, otherwise returns 0.
ceiling(X)	Smallest integer not less than X.
cos(X)	Cosine of X.
cosh(X)	Hyperbolic cosine of X.
cot(X)	Cotangent of X.
coth(X)	Hyperbolic cotangent of X.
csc(X)	Cosecant of X.
csch(X)	Hyperbolic cosecant of X.
digstate()	Returns a system digital state value. • If successful, the system digital state is returned as a system enumeration value. For PI AF Client versions older than 2.4, the value is returned as a PI SDK digital state, because the system enumeration value set did not exist. • If unsuccessful, the value is returned as a string. In both cases, the `AFValue.IsGood` flag is set to false, and the attribute is flagged in the Attributes Viewer with the icon.
e()	Value of the natural logarithm base.
exp(X)	Natural logarithm base e raised to power of X.
floor(X)	Largest integer not greater than X.
ln(X)	Natural logarithm of X.
log(X)	Logarithm of X using base 10.
logbase(X,Y)	Logarithm of X using base Y.
max(X,Y)	Greater of X and Y.
min(X,Y)	Lesser of X and Y.
normalrnd(X,Y)	Random number that maps the normal distribution curve. X is the mean and Y is the standard deviation.
pi()	Value of pi.
poisson(X)	Random number that maps the Poisson distribution. X is the mean.
pow(X,Y)	X raised to the power of Y.
rand()	Uniform random number. The values can be between 0 and less than 1.0
rand(X,Y)	Uniform random number. The values can be between X-Y/2 and less than X +Y/2.
remainder(X,Y)	Returns the remainder resulting from the division of X by Y.
round(X)	X rounded to the nearest whole number.
roundfrac(X,Y)	X rounded to the number of fractional digits specified by Y. Y is an integer number.
sec(X)	Secant of X.
sech(X)	Hyperbolic secant of X.
sign(X)	Returns 1 if X is greater than zero and -1 if X is less than zero.
sin(X)	Sine of X.
sinh(X)	Hyperbolic sine of X.

Function	Description
sqrt(X)	Square root of X.
tan(X)	Tangent of X.
tanh(X)	Hyperbolic tangent of X.

Units of measure in formula data references

When working with formulas and units of measure, you need to configure the equation inputs and the output with the correct UOM:

- For the attributes in the equation, use the UOM that the formula expects (NOT the units the attribute is already in).
- For the calculation result, specify the UOM in which you want the result to appear. This must be consistent with the UOMs for your inputs (attribute values).

Example of UOM in a formula data reference

Consider the following formula configuration for converting volume and density into mass:

```
V=Volume;UOM=L;D=Density;UOM=kg/L;[V*D];UOM=kg
```

The units of measure for the inputs and outputs are consistent: (L*kg/L = kg). This formula works on any input attributes or output attributes, regardless of the attribute's unit of measure, as long as they have a unit of measure of the correct class specified.

Configure formula data references

Create formula data references to perform custom calculations.

Before you start

The formula data reference does not support strings or other non-numeric value types.

Procedure

1. Choose one of the following actions.
   - In the Elements browser, select the desired element.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. From the **Data Reference** drop-down list, choose **Formula**.
4. Click the **Settings** button to open the Formula Data Reference configuration window.
5. If your equation requires an attribute value, click to add it as a parameter (Define parameters for formula data references).
6. Select the **Default Values Allowed** check box to enable PI AF to use attribute default values in the calculation. Default values are specified on the attribute template. If you clear the check box, the calculation fails when the data for one or more attributes is not available.

7. Click to begin defining the equation. For more information, see Define equations for formula data references.
8. In the **Unit of Measure** field, specify the unit of measure the result of the formula or calculation sequence will produce, as described in Units of measure in formula data references.
9. In the **Minimum** field, enter an appropriate value for the minimum returned value. If the calculated value is less than this minimum, the data reference returns the minimum value you specify. If there is no minimum value for this calculation, leave the **Minimum** box blank.
10. In the **Maximum** field, enter an appropriate value for the maximum returned value. If the calculated value is greater than this maximum, the data reference returns the maximum value you specify. If there is no maximum value for this calculation, leave the **Maximum** box blank.

    Note:
    Select the **Stepped** check box for the value to be stepped when plotted.

11. Click **Evaluate** to test the data reference. The value returned by the calculation sequence appears in the adjacent box. When the formula data reference is configured under a template attribute, the calculation uses the default values for the template attributes. When the configuration is done under an element attribute the actual data is used for the calculation.
12. Click **OK**.
13. Configure the rest of the attribute settings as you would for any other attribute.

Topics in this section

- Define parameters for formula data references
- Define equations for formula data references

Define parameters for formula data references

When equations require attribute values, add those attributes as parameters in the Formula Configuration window.

Procedure

1. Click next to the **Parameters** field.

   Note:
   You can alternatively click in an empty row and enter a variable (a -z).

2. In the Parameter Configuration window, select a variable from the **Variable** menu. The number of variables available is limited to 26 chars (from A to Z).
3. Select the attribute that the variable represents.
   - Choose the attribute from the **Attribute** menu. All sibling attributes with supported value types are listed.
   - If the attribute that you want is not in the menu, choose **Other**. A new field called **Attribute** appears in the window. Click to pick your attribute from the tree.

Alternatively, type in the path to the attribute of interest. Examples of some attribute paths are listed.

4. Select the desired unit of measure for the attribute. Note that this is not the UOM of the attribute but rather the UOM that your formula requires from this input, as described in Units of measure in formula data references.

   For example, if your input attribute has a UOM of Celsius but your equation requires Fahrenheit, you would select Fahrenheit in this field.

Define equations for formula data references

Define equations for use in formula data references.

Procedure

1. To add a new formula row, click next to the **Equations** field. You can also click in the text area and type in an equation.
2. Type an equation directly or build your equation from a list of available variables, operators, functions, and substitution parameters. Click to choose from the following actions:

To ...	Do this ...
Enter a variable	Click **Variables** and click a letter on the alphabetized list.
Enter an operator	Click **Operators** and click an operator on the list. Valid operators are listed in Formula data reference operators.
Enter a function	Click **Functions** and click a function on the list. Valid functions are listed in Formula data reference functions.
Enter a substitution parameter	Click **Substitution Parameters** and click a time substitution parameter. For elements and transfers, only **`%Time%`** is available, but for event frames you can select from **`%StartTime`**, **`%EndTime%`**, **`%UtcTime%`**, **`%UtcStartTime%`**, **`%UtcEndTime%`**, and **`%Duration%`**. For more information on these parameters, see List of PI AF substitution parameters. Numbers are converted to a double, time spans to a duration in seconds, and timestamps to UTC seconds.

3. To add another row to the formula, click and continue the formula. Repeat as needed until you have completed the formula.
4. To change the formula, choose from the following actions:

To ...	Do this ...
Remove the selected equation from the calculation sequence	Select the row you wish to remove and click .
Remove all equations from the calculation sequence	Click .
Move an equation up or down in the calculation sequence	Select the row to be moved and click or .

String Builder data references

The **String Builder** data reference enables you to apply string manipulation functions, such as concatenation, to an attribute's values, and output a reformatted string. This is useful when you need to obtain a string or numeric value type from other element attributes. String Builder data references do not perform Unit of Measure conversions or calculations. When you use the **String Builder** data reference in a template, value substitution takes place at run time.

Expression syntax

Observe the following rules as you build an expression:

- You can construct an expression on a single line, using semicolons to separate its terms.
- You can also place each term on its own line, which eliminates the semicolons and makes the expression structure more apparent.
- Enclose substitution parameters in double quotes so that their result is treated as a literal string.
- You can include any keyboard character in a double-quoted string, for example, dashes, backslashes, colons, semicolons, and so on. To include the double-quote character in a string, type two consecutive double-quote characters within the string, for example, `"""t""e""s""t"""` produces `"t"e"s"t"`.
- You can include single-quoted attribute paths to display the value of the referenced attribute in the result.
- You should avoid the use of unquoted strings because you may get haphazard results.

Topics in this section

- Attribute references in String Builder
- Function implementation in String Builder
- Create String Builder data references

Attribute references in String Builder

To reference other attributes, you use the syntax in the following table.

Object	Syntax
Sibling attribute	`siblingAttribute`
Top level attribute of same element	`|topLevelAttribute`
Parent attribute	`..`
Parent element attribute	`..\\|myAttribute`
Child attribute	`.|childAttribute`
From event frame to primary element	`.\Elements[.]|myAttribute`
Database relative path	`\TopLevelElement|myAttribute`
Full path	`\\myServer\myDatabase\myElement|myAttribute`

Examples

Suppose you want to retrieve the `Environment` PI point value from the `Cracking Process` parent element:

In the child element `Equipment`, you would use the syntax `..\|myAttribute` and enter `'..\|Environment'` in the String Builder Data Reference window:

A value similar to the following is displayed in the **Value** field: **43.3071403505214**.

Function implementation in String Builder

In String Builder, you can use several text manipulation functions, as well as the **`Format`** function.

Text extraction and case manipulation

You can use functions to manipulate the case of a string and to extract certain sections.

The syntax for text manipulation functions in String Builder is described in the following table:

`Left(string, length)`	Returns a string that contains the specified number of characters (`length`) from the left of the input (`string`). Example: `Left("Temperature",4)` returns `Temp`
`Right(string, length)`	Returns a string that contains the specified number of characters (`length`) from the right of the input (`string`). Example: `Right("GasTemp",4)` returns `Temp`
`Mid(string, start, [length])`	`length` is optional. Returns a sub-string from the specified position (`start`) of the input (`string`). When number of characters (`length`) is included, returns the specified number of characters. Example: `Mid("GasPressure",4,8)` returns `Pressure`

UCase(string)	Converts `string` to uppercase. Example: `UCase("Temperature")` returns `TEMPERATURE`
LCase(string)	Converts `string` to lowercase. Example: `LCase("TEMPERATURE")` returns `temperature`
Trim(string)	Removes blanks on both sides of `string`. Example: `Trim(" Temperature ")` returns `Temperature`
RTrim(string)	Removes trailing blanks from `string`. Example: `RTrim(" Temperature ")` returns `" Temperature"`
LTrim(string)	Removes leading blanks from `string`. Example: `LTrim(" Temperature ")` returns `"Temperature "`
Replace(string, string, string)	In original `string`, replaces specified `string` with new `string`. Example: `Replace("Temperature","Temp","External Temp")` returns `"External Temperature"`

You can also nest text manipulation functions.

- For example, for an attribute named `GasPressure`, you can use the **`Mid`** function in combination with the **`UCase`** function to return the following expression: `GASPR`
  ```
  Mid(UCase("%Attribute%"), 1, 5);
  ```
- Alternatively, for the same attribute, you can use the **`Mid`** function in combination with the **`LCase`** function to return the following expression: `pressure`
  ```
  Mid(LCase("%Attribute%"), 4, 8);
  ```

Note:
Beginning with PI AF 2018, you can nest functions in any position, not the first position only. For example,

```
Replace(Mid("abc",1,3),"b","2");
Replace("abc",Mid("abc",2,1),"2");
Replace("abc","b",Mid("123",2,1));
```

all return the expression: `a2c`

Format function

The **`Format`** function enables you to convert real numbers, integers, and time stamps to a string according to the format and optional culture specification.

Note:
Unlike other string functions, the syntax of **Format(DateTime, ...)** follows C# syntax.

The syntax for **`Function`** implementation in String Builder is described in the following table:

Real numbers	**`Format(real, format)`** **`Format (real, format, culture)`**	Format follows Performance Equation (PE) style syntax, such as `"%0.3f"`, where the number before the decimal indicates the minimum total number of characters to output, pre-padding with blanks, and the number after the decimal indicates the number of digits to display after the decimal point. Culture is optionally specified using an IETF language tag, such as `"en"`, `"en-US"`, `"de"`, which specifies the language and optional culture and regions.
Integer numbers	**`Format(integer, format)`** **`Format(integer, format, culture)`**	Format follows Performance Equation (PE) style syntax, such as `"%4d"`, where the number indicates the minimum total number of characters to output, pre-padding with blanks as necessary. Culture is optionally specified using an Internet Engineering Task Force (IETF) language tag, such as `"en"`, `"en-US"`, `"de"`, which specifies the language and optional culture and regions.
Time stamps	**`Format(datetime, format)`** **`Format(datetime, format, culture)`**	Format follows C# `DateTime.ToString(format)` syntax, and can be either a pre-defined syntax, or a custom syntax. `DateTime` format uses invariant culture settings. To display dates and times for a specific culture, add an Internet Engineering Task Force (IETF) language tag, such as `"fr-FR"` and `"fr-CA"`, as described in the MSDN article Table of Language Culture Names, Codes, and ISO Values Method (https://msdn.microsoft.com/en-ca/library/ee825488(v=cs.20).aspx). For more information on date and time formats, see Format strings for time substitution parameters.

Numeric format example

Suppose you want to format the `Environment` PI point value that you have already retrieved from the `Cracking Process` parent element. In the String Builder Data Reference window, you would select the row that contains the string expression, click ▸ and select **Functions** > **Format(Real, "%3.3f")** to modify the expression:

The **Value** field changes from `43.3071403505214` to `43.307`.

To change the cultural value from the US default to a Spanish culture format, you would select **Functions** > **Format(Real, "%3.3f", "en-US")** and change the `"en-US"` string to `"es"`:

The **Value** field changes from `43.307` to `43,307`.

> **Note:**
> Be sure the data type you specify matches the attribute data type. You would encounter errors, for example, if you specify `"%3.2f"` for an integer type attribute value or `"%3d"` for a floating point attribute value.

Date time format example

Suppose you want to retrieve and format the `In Service Date` value from the `Condenser` child element:

In the parent element `Power and Steam Generation`, you would create an attribute named `Condenser Service Date`. Use the syntax `.\childElement|Attribute`, and enter `'.\Condenser|In Service Date'` in the String Builder Data Reference window:

The `In Service Date` value is retrieved and displayed in the **Value** field: `2/25/2009 12:00:00 AM`.

To change the format from the US default to Universal full date in German culture format, you would alter the expression to read `Format(.\Condenser|In Service Date,"U","de")`. The `Condenser Service Date` attribute value format is converted accordingly:

Create String Builder data references

Create a String Builder data reference when you need to apply string manipulation functions, such as concatenation, to an attribute value and output a reformatted string.

Before you start

When a String Builder data reference is used in a template, value substitution takes place at run time.

Procedure

1. Choose one of the following actions:
   - In the Elements browser, select the desired element.
   - In the Event Frames browser, select the desired event frame.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. In the attribute configuration panel, choose **String Builder** from the **Data Reference** drop-down list.
4. Click **Settings**.
5. In the String Builder Data Reference window, click .
6. In the highlighted row, press F2 or click and select from the following options:

Option	Description
Literals	A text string between double quotation marks: "Sample"
Attribute Values	A list of other attributes for the selected element or element template. Note that attributes must appear in single quotes.
Related Attribute Values	A list of the following attribute references: ◦ '\\<Server>\<Database>\<Element>|<Attribute>' ◦ '\\.\<Database>\<Element>|<Attribute>' ◦ '\<Root Element>|<Attribute>' ◦ '.\<Child Element>|<Attribute>' ◦ '..\\|<Primary Parent Attribute>' ◦ '..\<Sibling Element>|<Attribute>' You can also select **Search** to enter search criteria in the Attribute Search window.
Substitution Parameters	A list of 20 commonly used substitution parameters. For more information, see Substitution parameter list.

Functions	A list of the following functions: ◦ Left(string, length) ◦ Right(string, length) ◦ Mid(string, length) ◦ Mid(string, start, length) ◦ UCase(string) ◦ LCase(string) ◦ Trim(string) ◦ RTrim(string) ◦ LTrim(string) ◦ Replace(string, string, string) ◦ Format(Real, "%3.3f") ◦ Format(Real "%3.3f", "en-US") ◦ Format(DateTime, "yyyy-MM-dd HH:mm:ss") ◦ Format(DateTime, "yyyy-MM-dd HH:mm:ss", "en-US") For examples of how these functions can be used, see Function implementation in String Builder.

7. Build your expression further. You can either type a semicolon (`;`) and continue in the same row, or click to continue on another row. You can use the following icons to manipulate the expression:

To ...	Do this ...
Append a new string	Click .
Remove a string	Select a row and click .
Remove all strings	Click .
Move a string up a row	Select a row below the top row and click .
Move a string down a row	Select a row above the bottom row and click .

As you build the string expression, you can preview the result of the expression in the **Value** field.

8. When the string expression is complete, click **OK**.

Examples

- You can enter the following substitution parameters in a single row to concatenate a pathname string. For example:

```
"%Database%";"\";"%Element%";"|";"%Attribute%";
```

This data reference would produce output similar to: `DB26\WX12000|pressure`.

- You can build an expression in separate rows to show the duration of an event. For example:

a. Click , press F2 and select **Substitution Parameters** > **"%StartTime%"**.

b. Click and type the characters: `" to "`

c. Click , press F2 and select **Substitution Parameters** > **"%EndTime%"**.

The **Value** field shows a date and time interval:

Table lookup data references

You can configure an attribute to obtain its value from a PI AF table. You can define the PI AF table entirely in PI AF, or you can link to or import from a data source outside the PI System (such as a Microsoft SQL table or a Microsoft Excel worksheet).

Table lookup

The table lookup data reference is intended to be a simple table lookup. It uses Microsoft's .NET **`System.Data.DataTable.Select`** method, and follows the filter expression rules of that method. Note that not all its features are implemented in PI AF. For example, a table lookup data reference does not include functionality such as column functions and parent/child relation referencing.

Note:
The table lookup data reference is not optimized to treat data in an external table as a time series and query for it. For optimal performance, such data should be stored in PI tags on a PI Data Archive.

Table lookup methodology

You begin by creating a table profile in the Library browser, where you establish general settings for the time zone and cache interval for refreshing data, as well as definitions of table columns and connections to data sources outside the PI System if you are linking to or importing external data.

If you are linking to external data sources, consider creating reusable table connection profiles in the Library browser, as described in Create reusable table connections.

Videos

For information on how to create internal tables for table lookup data references, watch this video:

https://www.youtube.com/watch?v=3gQzzNWzd6o&rel=0

For information on how to configure table lookup data references, watch this video:

https://www.youtube.com/watch?v=hoy7QarxF1o&rel=0

Topics in this section

- Create PI AF tables
- Create table column definitions
- Populate tables manually
- Data references from outside the PI System
- Configure table lookup data references

Create PI AF tables

You create a table profile so that attributes can obtain their values from an internal or external table.

Procedure

1. In the Library browser, right-click the **Tables** collection and click **New Table**.
2. In the **General** tab, enter a **Name** and **Description** for the table.
3. Optional. Assign one or more categories to the table. Enter table category names directly, separated by a colon, or click and select them from the Categorize window.
4. Decide how `DateTime` values should be stored in the table. By setting the value for the table overall, you need not specify it every time you create a table lookup data reference.
   a. In the **Time Zone** field, select a time zone from the list.
   b. Select or clear the **Convert To Local** check box. Choose from the following options:
   - To convert `DateTime` values to the local time, select the **Convert To Local** check box.
   - To display `DateTime` values in the time zone selected in the **Time Zone** field, clear the **Convert To Local** check box.

   Note:
   When a client application queries the table, the time zone in which `DateTime` values are displayed depends on whether the **Convert To Local** check box is selected. PI System Explorer uses this setting to determine the precise `DateTimeMode` to use in a Microsoft .NET data table, as described in Conversion settings for time zones.
5. In the **Cache Interval** field, enter the amount of time until the table's cached data is automatically refreshed. From the list, choose whether the value is in seconds, minutes, hours, or days. The default value is zero, which indicates a manual refresh.

> **Note:**
> Automatic refreshing is disabled if the table has changes that have not been saved to the server.

> **Note:**
> With manual refreshes, you refresh table data by right-clicking the table in the Library browser and clicking **Refresh**. Otherwise, table data that is being queried by client applications is refreshed only when the application is restarted. For example, when PI System Explorer is opened or whenever the Microsoft Internet Information Services (IIS) application pool recycles the client (the default is every 29 hours).

6. Check in your work.

> **Note:**
> The **Connection** and **Query** fields are read-only. PI AF populates these fields when you link to or import an external table.

After you finish

Define and populate the table in one of three ways:

- Manually define and populate the table in PI System Explorer. First, create column definitions on the **Define Table** tab, as described in Create table column definitions. Next, enter data on the **Table** tab, as described in Populate tables manually.
- Import a table from outside the PI AF server. See Access to Microsoft Access or Excel data.
- Link to a table outside the PI AF server. See Access to Microsoft Access or Excel data.

Conversion settings for time zones

The precise time conversion calculation depends on:

- The value that you select in the **Time Zone** list. Times are prefixed with either Universal Coordinated Time (UTC), Greenwich Mean Time (GMT), or Time Zone (TZ). The list also includes the None option.
- Whether you selected or cleared the **Convert to Local** check box.
- Whether this is an internal (either imported or defined in PI AF) or external (linked) table.

The following table explains how displayed times are determined for each possible combination. The exact options in the **Time Zone** list are dependent on your operating system.

Time zone	Convert to local	Internal/ external	Behavior
None	✓	Internal	Fields that contain `DateTime` data type are automatically adjusted for time zone differences and return local times on the client.
		External	Fields that contain `DateTime` data type are assumed to be in the time zone of the PI AF server and are adjusted to the client's local time.

Time zone	Convert to local	Internal/ external	Behavior
None	✗	Internal	Fields that contain `DateTime` data type are not translated. Useful for storing date fields where translation is not desired, such as birth date.
		External	Fields that contain `DateTime` data type are not translated. Useful for storing date fields where translation is not desired, such as birth date.
`UTC` or `GMT`	✓	Internal	Fields that contain `DateTime` data type are adjusted to local times on the client.
		External	Fields that contain `DateTime` data type are externally stored as UTC, but adjusted to local time on the client, represented as UTC.
`UTC` or `GMT`	✗	Internal	Fields that contain `DateTime` data type are always represented as UTC.
		External	Fields that contain `DateTime` data type are externally stored as UTC, and left as UTC on the client.
`TZ`	✓	Internal	Fields that contain `DateTime` data type are adjusted for time zone differences and are local time on the client. This combination is not normally needed except when external data is being imported into PI System Explorer.
		External	Fields that contain `DateTime` data type are adjusted for time zone differences, and are adjusted to local time on the client, represented as UTC.
`TZ`	✗	Internal	Fields that contain `DateTime` data type are left in the time zone specified. A table lookup data reference adjusts as appropriate; however, other applications may not.
		External	Fields that contain `DateTime` data type are left in the original time zone specified, represented as UTC. A table lookup data reference adjusts as appropriate; however, other applications may not.

Create table column definitions

To create a table manually, begin by defining the columns in the table.

Procedure

1. Once you have completed the overall table settings on the **General** tab, click the **Define Table** tab.
2. Determine the number of columns in the table and click once for each column to be defined.
3. Click the **Name** cell and replace the default label with the required column name.
4. Click the adjacent **Value Type** cell and select a data type from a list of **Basic Types** or **Array Types**. For a description of data types, see PI AF data types.

> **Tip:**
> If any value type is set to `DateTime` or `String`, OSIsoft recommends that you leave the column time zone value set to **None** and select a time zone value for the table overall in the **Time Zone** field on the **General** tab. That value supersedes time zones set on specific table columns.

5. Optional. Click the **Unit of Measure** cell and choose one of the following actions:
   - Start typing and select a unit of measure from the list of matching UOMs.
   - Click ˅ and select from the full list of UOMs.

> **Tip:**
> OSIsoft recommends that you set a unit of measure for a table column so that you do not need to specify one each time you select the column in a table lookup data reference.

6. Repeat steps 3 to 5 for each column in the table.
7. To modify the table column definition, right-click a row in the grid and select an option. Alternatively, use the following icons.

To ...	Do this ...
Insert a new column	Click .
Move a column up a row	Select a row below the top row and click .
Move a column down a row	Select a row above the bottom row and click .
Remove a column	Select a column and click .

After you finish

When you have completed defining table columns, click the **Table** tab so you can enter row data for each column manually.

Populate tables manually

Once you have completed table column definitions, enter row data for each column.

Procedure

1. Click the **Table** tab. Based on the table columns that you created in the **Define Table** tab, enter the appropriate information in rows for each column.
2. Choose from the following actions:

To ...	Do this ...
Enter a new row	Right-click in the table grid and click **Insert** or **New**.
Copy and paste rows from an Excel spreadsheet	a. In the spreadsheet, copy the rows you want. b. In the table grid, right-click a new row or a range of rows and click **Paste**.

Replace data for specific cells	a. Use standard Windows selection keystrokes (such as SHIFT+<click> and CTRL+<click>) to select contiguous and non-contiguous cells in the table, as described in Select multiple objects in the viewer. b. Right-click and click **Clear Cell(s)**. c. Enter new data in the cleared cells.
Remove a row	Right-click the row you want to remove and click **Delete**.

3. To save your work, press CTRL+S or click **Check In**.

Data references from outside the PI System

You can also use PI AF tables to access data that is external to the PI System. Such data might be contained in Microsoft Excel, Access, or SQL Server, or other OLE DB/ODBC data sources. You can either import the table or link to it after you have defined the table structure, as described in Create PI AF tables and Create table column definitions.

Imported tables

PI AF tables with imported data are called imported tables. Imported tables are read/write tables. They are limited in size but are more secure than linked tables. Imported tables are sometimes called internal tables because, unlike linked tables, the table data is managed in PI AF. After the initial import, there is no further relationship between the foreign table and the PI AF table. You can edit the data directly in PI AF.

It is a good practice to limit your imported tables to 10,000 rows of data or less. Imported tables are not designed for storing very large databases. If you need to access a lot of data in PI AF tables, link to external tables instead, which do not present such storage limits. Alternatively, break the table into separate tables when importing.

Linked tables

Linked tables are sometimes called external tables, because the source data is not stored in the PI AF database. You cannot edit an external table from PI AF. Linked tables require additional security configuration because you need to configure how PI AF connects to the external data source. You should set up a reusable table connection, where you configure the type of authentication to access the external table.

Videos

For information on how to import data from external tables, watch this video:

https://www.youtube.com/watch?v=MmXmnac969s&rel=0

For information on how to link to data in external tables, as well as SQL security, watch this video:

https://www.youtube.com/watch?v=JnhodkkyHks&rel=0

Topics in this section

- Authentication for linked tables
- Create reusable table connections
- Access to Microsoft Access or Excel data

- Access to SQL Server data

Authentication for linked tables

When a client application requests external data, the PI AF server queries the external data source and returns the data to the client as a read-only PI AF table.

For externally linked tables, OSIsoft recommends that the OLE DB provider and the PI AF Server have the same bitness (32-bit or 64-bit). To configure an external table connection in PI System Explorer, for example, you would use a PI AF server of the same bitness (typically, 64-bit).

When you configure the linked table, you must specify the credentials that the PI AF server uses to connect to the database. The authentication options are:

- **Impersonate Client**

  If the source database supports Windows authentication, use the Windows identity of the client that is requesting the data. This is an impersonated connection. This is the most secure method of authentication and should be used wherever possible.

- **Supply Password**

  If the source database does not support Windows authentication, or if the database and PI AF server are on different, non-trusted domains, specify a user name and password with the necessary access on the source database. PI AF uses this hard-coded account to read the data in the external data source. For example, MySQL database does not support Windows authentication, so you would use the user name and password of an account on the MySQL database.

  Note:
  You can enter a user ID and password as part of the connection string and save it with a PI AF table connection, regardless of whether support for external PI AF tables for non-impersonated users has been previously enabled (with the `afdiag /DTImp` command).

- **No additional security context**

  This option usually applies when you use Excel or other file-based data sources; otherwise every user needs to be granted read access to the file on the server. With this option, the external table will be accessed using the PI AF Server's identity. In this case, you do not need to specify a username or password when configuring the linked table, nor is Kerberos configuration required.

  Caution:
  Take care to configure SQL security in such a way that the PI AF server's identity does not have more privilege than necessary to retrieve the data. Only PI AF administrators are allowed to configure external tables for security reasons. For that reason, ensure that the PI AF `Administrators` identity and the **Admin** access right are assigned to only a limited set of users when this connection mode is enabled.

Topics in this section

- Restrictions on non-impersonated connections
- Risk of using non-impersonated connections
- Data access recommendations for linked tables

- Security settings for linked tables

Restrictions on non-impersonated connections

Because there are security risks for linked tables that use non-impersonated connections, some PI AF server system administrators restrict or prevent their use. Your system administrator might:

- Require administrative privileges on the PI AF server and write privileges on the PI AF table.
- Prevent creation of linked PI AF tables with non-impersonated connection.
- Prevent creation of any linked tables.

If you have problems with linked tables, consult your system administrator about the PI AF server external table settings.

Risk of using non-impersonated connections

Depending on the configuration of the SQL Server, a user with PI AF administrator privileges could create attacks on the SQL Server and take full control of the system if these following conditions exist:

- A PI AF table is configured to use the PI AF server identity for linking to an external database.
- Non-impersonated linked (external) tables are enabled on the PI AF server.

  By default, non-impersonated linked tables are disabled on the PI AF server. In order for a user to execute an attack, that user would need to enable non-impersonated external tables.
- The PI AF server account has administrative rights on a SQL Server.

  By default, the PI AF server runs under a virtual account, `NT SERVICE\AFService`, and does not have administrative rights to the locally-configured SQL Server or access to remote computer databases. Without administrator rights to the remote database, the possibility for elevation of privilege attacks is limited.

> **Caution:**
> For security reasons, do not grant the PI AF server administrative privileges on the computer or SQL Server when running with non-impersonated queries.

Data access recommendations for linked tables

OSIsoft recommends you observe the following guidelines when you set up linked tables.

- If access to linked tables is not needed, disable it altogether.
- Do not grant the PI AF application service account administrative privileges on the PI AF server or SQL Server when running with non-impersonated queries.
- You must have administrative privileges on the PI AF server to configure an external table that runs non-impersonated queries.

Security settings for linked tables

You use the PI AF Diagnostics utility (**afdiag** located in the `\PIPC\AF` folder) to enable or disable support for external PI AF tables. Since the utility makes a direct connection with the associated SQL Server database, the SQL Server `sysadmin` or `db_AFadmin` role is required.

Use the PI AF Diagnostics utility to adjust security settings for external tables.

Task	Command	Default Setting
Enable support for external PI AF tables	**afdiag /DT**	enabled
Disable support for external PI AF tables	**afdiag /DT-**	
Enable support for external PI AF tables for non-impersonated users	**afdiag /DTImp**	disabled
Disable support for external PI AF tables for non-impersonated users	**afdiag /DTImp-**	
Change security settings for a specific PI AF table	In the Library browser, right-click on the table and click **Security**.	By default, table configuration requires administrative privileges on the PI AF server.
Change security settings for all tables	In the Library browser, right-click on **Tables** and click **Security**.	By default, table configuration requires administrative privileges on the PI AF server.

Note:
You do not have to enable support for external PI AF tables for non-impersonated users. You can include a user ID and password as part of the connection string in a PI AF table connection and check it in, regardless of whether support for external PI AF tables for non-impersonated users is enabled. The defined PI AF table connection can be used within a PI AF table definition. This means that if a SQL Server Login has permission to access the data referenced in the connection string, a PI AF table linked to that PI AF table connection can retrieve the external data.

Create reusable table connections

Create an OLE DB connection and reuse it to configure linked tables from the same data source.

Before you start

Use the 64-bit PI System Explorer to configure connections to linked tables. It includes 64-bit OLE DB drivers, which are required for the 64-bit PI AF Server.

Procedure

1. In the navigator, click **Library**.
2. Choose from the following actions:
   - Select **Table Connections** and click **New Table Connection** on the toolbar.
   - From the browser, right-click **Table Connections** and click **New Table Connection**.

Table connection properties are displayed in the viewer with a default name in the **Name** field.

3. Optional. Edit the default name and add a description for the table connection in the **Description** field.
4. In the **Connection** field, you can either enter a connection string directly, or click **Build** to configure the connection in the Data Link Properties window.
5. In the **Provider** tab of Data Link Properties, choose an OLE DB provider and follow onscreen instructions to configure the connection.

   For help on Microsoft Data Link, click **Help** from any tab, link to the MSDN Resources page, and enter `Data Link Properties Dialog Box` in the search field.
6. In the viewer, select a **Security** option. Some options may not be available.
   - **Impersonate Client** (recommended)
   - Select **Supply Password** and enter a password for the table connection.
   - **No additional security context**

   See Authentication for linked tables for information on these options.
7. On the toolbar, click **Check In** to check in and save the table connection.

Access to Microsoft Access or Excel data

To access data from a Microsoft Access database or a Microsoft Excel workbook, you can link or import the data. The data in a linked table is refreshed when the table is accessed and whenever the time since the last refresh exceeds the table's **Cache Interval** setting.

Imported data is loaded into the PI AF table once. If you ever need to refresh the imported data in a table, you can right-click the table in the Library browser and click **Re-Import Table**.

You can link to or import data from a Microsoft Access table or a Microsoft Excel workbook after you perform the following steps:

- Specify the source database or workbook.
- Create a query that returns the desired data.
- Enter any login credentials required to access the database or workbook.

The exact instructions depend on your hardware configuration and what you are trying to do:

32-bit PI AF server	Follow the instructions in Link or import data.
64-bit PI AF server	• To import, follow the instructions in Link or import data. • To link, follow the instructions in Link to data on a 64-bit PI AF server.

Topics in this section

- Link or import data
- Link to data on a 64-bit PI AF server

Link or import data

These instructions describe how to import data on a 32-bit or 64-bit PI AF server and how to link to data from a 32-bit PI AF server only. To link to data from a 64-bit server, see Link to data on a 64-bit PI AF server.

Procedure

1. In PI System Explorer, navigate to the PI AF table or create one as described in Create PI AF tables.
2. In the **Library** pane, expand the **Tables** node, and click the desired PI AF table.

   The table details display in the right pane.
3. Click **Link** or **Import**.

   The corresponding window opens.
4. Link only: If you are linking the table, enable the **Impersonate Client** option (not displayed for Import).
5. Click **Build**.

   The Data Link Properties window opens.
6. On the **Provider** tab, select the provider according to the version of Microsoft Office that you are using and click **Next.**
   - Office 97-2003: select **Microsoft Jet 4.0 OLE DB Provider**.
   - Office 2007 and higher: select **Microsoft Office 12.0 Access Database Engine OLE DB Provider**.
7. On the **Connection** tab, specify the following and click **OK**.
   - **Data Source**

     The location and file name of the database or workbook (such as `C:\AFTestData.xls`). If you are linking, the path to the file must be relative to the PI AF server.
   - **User Name**

     Login credentials of a user that has been granted read access to the database or workbook.

   > **Note:**
   > To store the password with the connection information, select the **Allow Saving Password** check box. The password is stored as plain text (not encrypted).
8. On the **Advanced** tab, in the **Access permissions** list, select **Share Deny None.**
9. Excel only: On the **All** tab, select the **Extended Properties** value and click **Edit Value**.

   The Edit Property Value window opens.

   Enter the property value according to the version of Microsoft Excel that you are using, and then click **OK**.
   - Excel 97-2003: Excel 8.0
   - Excel 2007 and higher: Excel 12.0

10. To verify that the spreadsheet is accessible, return to the **Connection** tab and click **Test Connection**.

    If the settings are valid, a Test connection succeeded message displays.
11. To dismiss the window and return to PI System Explorer, click **OK**.
12. To define the data to be returned from the spreadsheet, enter an SQL query in the **Query** field. To dismiss the window, click **OK**.
    - Microsoft Excel example: `SELECT * FROM [Sheet1$]`
    - Microsoft Access example: `SELECT * FROM Table1`
13. To review the resulting data, examine the **Table** tab. If the query is specified correctly, the tab contains a table displaying the results.
14. To save your changes, right-click the table node and choose **Check In**.

Link to data on a 64-bit PI AF server

To link to data in an Access database or Excel workbook on a 64-bit PI AF server, you must use the 64-bit Access Database Engine (ACE) data provider; there is no 64-bit Jet data provider.

Procedure

1. In PI System Explorer, navigate to the PI AF table or create one as described in Create PI AF tables.
2. In the **Library** pane, expand the **Tables** node, and click the desired PI AF table.

   The table details display in the right pane.
3. Click **Link**.

   The corresponding window opens.
4. Enable the **Impersonate Client** option.
5. In the **Connection** field, enter a valid connection string for the Excel workbook or Access database, using the Microsoft Office 12.0 Access Database Engine OLE DB Provider (must be installed on the PI AF server), as in the following examples:
   - Microsoft Excel example:
     ```
     Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\example.xlsx;Extended
     Properties="Excel 12.0 Xml";
     ```
   - Microsoft Access example:
     ```
     Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\example.accdb;Persist
     Security Info=False;
     ```
6. To define the data to be returned from the spreadsheet, enter an SQL query in the **Query** field. To dismiss the window, click **OK**.
   - Microsoft Excel example: `SELECT * FROM [Sheet1$]`
   - Microsoft Access example: `SELECT * FROM Table1`
7. To review the resulting data, examine the **Table** tab. If the query is specified correctly, the tab contains a table displaying the results.
8. To save your changes, right-click the table node and choose **Check In**.

Access to SQL Server data

When you import data from or link to data in a Microsoft SQL Server table, you must define valid connection information. The steps for linking or importing depend on the connection method that you choose, as described in Authentication for linked tables.

Topics in this section

- Use Windows impersonated security connection
- Use Windows non-impersonated security
- Use SQL Server security
- Configure security on the target table's database
- Import or link SQL server tables

Use Windows impersonated security connection

Link to or import data from a SQL Server table with the **Impersonate Client** option. See Data access recommendations for linked tables.

> **Note:**
> If linking to a SQL Server that is not on the same computer as the PI AF server, it may be necessary to configure Kerberos to allow the client's identity to be forwarded from the PI AF server to the SQL Server.

Procedure

1. Create a local user group for impersonated clients.
2. Configure security on the target table's database.
3. Import or link SQL server tables.

Create a local user group for impersonated clients

If the table to which you want to connect resides in a SQL Server instance other than the one where the PI AF SQL database (PIFD) resides, ensure that the table is accessible via Windows authentication.

Procedure

1. On the computer where the SQL Server instance resides, click **Start** > **Administrative Tools** > **Computer Management**.

   The Computer Management application starts.
2. Expand **Local Users and Groups**.
3. Right-click **Groups** and choose **New Group**.
4. In the New Group window, create a local user group for the users that require access to the database table.
5. Add to this group the accounts of all users that might be impersonated.
6. Click **OK** to add the selected users, and then click **Close** to dismiss the New Group window.
7. Close the Computer Management application.

Use Windows non-impersonated security

Link to or import from a SQL Server table with the `no additional security` context option.

Procedure

1. Create a local user group for PI AF application service account
2. Configure security on the target table's database
3. Import or link SQL server tables

Create a local user group for PI AF application service account

If the table to which you want to connect resides in a SQL Server instance other than the one where the PI AF database (PIFD) resides, ensure that the table is accessible to the PI AF application service account.

Procedure

1. On the computer where the SQL Server Instance resides, click **Start** > **Administrative Tools** > **Computer Management**.

   The Computer Management application starts.
2. Expand **Local Users and Groups**.
3. Right-click **Groups** and choose **New Group**.
4. In the New Group window, create a local user group to hold the identity of the PI AF Server Application Service.
5. Add the account of the user associated with the PI AF Server Application Service to the new group. If the PI AF Server Application Service is running under the `NT AUTHORITY\NetworkService` account, then add the PI AF server's computer account to this group.

   Note:
   If the PI AF Server Application Service is running as the Local System or Local Service account, you most likely need to use SQL Server authentication (SQL Server and Windows authentication mode) instead of Integrated security.
6. Click **OK** to add the selected user.
7. Close the Computer Management application.

Use SQL Server security

Link or import data from a SQL Server table using SQL Server authentication.

Before you start

If you are connecting to a remote SQL Server instance, ensure SQL Server is configured to accept Remote Connections.

If you are using a SQL Server account, ensure the SQL Server instance is configured to allow mixed mode authentication.

Procedure

1. Create a SQL Server user.
2. Configure security on the target table's database.
3. Import or link SQL server tables.

Create a SQL Server user

If the target table (the table to which you want to connect) resides in a SQL Server instance other than the one in which the PI AF database (PIFD) resides, create a user and enable database access for the user as described in this topic.

Procedure

1. Open Microsoft SQL Server Management Studio and connect to the SQL Server Instance that contains the target table.
2. Under the SQL Server Instance, expand the Security folder and then expand the Logins folder.
3. Create a new Login and enter a name in the **Login Name** field.
4. Select the **SQL Server Authentication** option.
5. Enter the password in the **Password** and **Confirm Password** fields.
6. From the **Default Database** list, select the database that contains the target table.
7. Select the **User Mapping** page.
8. Select the row for the Database that contains the target table.
9. Select the **Map** check box for the selected database.
10. Click **OK** to close the Login – New window and save the new Login.
11. Expand the **Databases** folder, then the folder for the target database, and grant the necessary permission to execute the query that will be used to the Login just created.

    For example, if the query that will be used is a SELECT statement that specifies a single table, expand the **Tables** folder for the target database, expand the **Tables** folder, then right-click the table to which the query refers and choose **Properties**.
12. In the Table Properties window, select the **Permissions** page, then the Login, then **Grant the Login the Select Permission**. Click **OK** to close the Table Properties window.
13. Close Microsoft SQL Server Management Studio.

Configure security on the target table's database

Any account under which the specified query might be executed needs to be granted permission to execute that query.

Procedure

1. Open Microsoft SQL Server Management Studio and connect to the SQL Server Instance that contains the target table.
2. Under the SQL Server Instance, expand the **Security** folder, and then expand the **Logins** folder.

3. Right-click the **Logins** folder and choose **New Login**.
4. Use the **Search** button to find the group created in the previous section and choose that group for the Login name.
5. Select the **Windows authentication** option, and select the database that contains the target table as the `Default database`.
6. Select the User Mapping page.
7. Select the row for the Database that contains the target table.
8. Select the **Map** check box for the selected database.
9. Expand the **Databases** folder, then the folder for the target database, and grant the necessary permission to execute the query that will be used to the Login just created.

   For example, if the query that will be used is a `SELECT` statement that specifies a single table, expand the **Tables** folder for the target database, expand the **Tables** folder, then right-click the table to which the query refers and choose **Properties**.
10. In the Table Properties window, select the **Permissions** page, search for and select **Login**, then **Grant the Login the Select Permission** and click **OK** to close the Table Properties window.
11. Close Microsoft SQL Server Management Studio.

Import or link SQL server tables

Whether you are importing from or linking to Microsoft SQL Server tables, the process is essentially the same. The following instructions describe how to link to an existing PI AF table using the PI System Explorer.

Procedure

1. To browse to the target PI AF table, display the **Library** pane, expand the **Tables** node, and click the desired table.

   Table properties display in the right pane.
2. Click **Link**.

   The Table Link window opens.
3. Click **Build**.

   The Data Link Properties window opens.
4. On the **Provider** tab, select **SQL Server Native Client**.
5. On the **Connection** tab, configure the SQL Server instance that contains the database to which you want to connect.
6. Configure authentication:
   - For Integrated security, select the `Use Windows NT Integrated Security` option.
   - For SQL Server security, select the `Use a Specific User Name and Password` option. Then, enter the SQL Server Login name in the **User Name** field and enter the password in the **Password** field.

> **Note:**
> Ensure that support for external PI AF tables for non-impersonated users has been enabled with the **`afdiag /DTImp`** command.

7. Select the `Select the Database` option and choose the database that contains the table to which you want to connect.
8. To verify that connection settings work, click **Test Connection**. If the settings are correct, PI System Explorer displays a success message.

> **Note:**
> Test Connection verifies that the account with which the PI System Explorer is running has access to the specified database. However, if you choose Integrated Security in the Table definition and choose the `no additional security context` option for table security, the account associated with the PI AF Server Application Service is used to connect when a user displays the data by viewing the **Table** tab.

9. Click **OK**.
10. In the **Query** field, specify the SQL query that returns the desired data and click **OK**.

    If the connection string requires a User ID and Password, click the supply password button and enter the SQL login password used to test the connection. This replaces the value in the **Password** field.
11. To display the data retrieved by the query, view the **Table** tab.

Configure table lookup data references

Create a table lookup data reference when you want an attribute to obtain its value from a table column.

Procedure

1. In the Elements or Library browser, select the desired element or element template.
2. In the viewer, select the attribute or attribute template for which you want a table lookup value.
3. In the **Data Reference** field in the palette, choose **Table Lookup** from the list.
4. Click **Settings**.
5. In the Table Lookup Data Reference window, select a previously defined table from the **Table** drop-down list. You can also choose from the following options:
   - Click (Manage Tables) to open a list of tables you can search or filter. To select a table, highlight it in the list and click **OK.**
   - Click (Table Properties) to view or edit properties for the selected table.
   - Click (Create New Table) to define a new table.
6. From the **Result column** list, select the column in the table from which you want to read the value.

> **Note:**
> Select the **Stepped** check box for the value to be stepped when plotted in a trend. With this setting, there is no interpolation between the table values.

7. From the **Unit of Measure** list, select the appropriate unit of measure in which the data in the result column is stored.

   Note:
   It is preferable to set the UOM directly in the table definition so that you do not need to specify it with each table lookup.

8. In **Time Zone**, you can define a setting if it has not previously been set in the table or column definition.

9. From the **Rule** list, choose an option:

   - `Select first row matching criteria`

     Use the **Order by** list to specify the sorting order. This order is used to select a row when more than one row matches the criteria. For more information, see Select first row matching criteria.

   - `Summarize all rows matching criteria`

     Select a summary operation from the **Summary** list to perform the selected operation on the selected column over the range of rows that match the criteria. For more information, see Summarize all rows matching criteria.

   - `Table provided time series data`

     Choose this option if the table has values with associated time stamps and you wish to treat these values as time series data. From the **Time Column** list, select the table column that contains the time stamps you want to use. Only columns with a value type of DateTime are listed. The `WHERE` clause is not required when you choose this option. For more information, see Table provided time series data.

     Note:
     To ensure that columns with DateTime value types are displayed, only specify time zone settings on the **General** tab of the overall table definition. Do not specify time zones on the column definition (on the **Define Table** tab).

10. In the **Where** pane, use the menus and buttons to build the table query.

    Note:
    You can manually type the entire clause into the **Complete WHERE Clause** text field. See WHERE clause syntax for more information.

    - From the **Column** list, select the column of the table to use in the query.
    - From the **Operator** list, select the relational operator to use in the query.
    - From the **Attribute or Value** list, select an attribute or a literal value to use in the query.
    - Click **Add And** or **Add Or** to write the `WHERE` clause into the **Complete WHERE Clause** field with an `AND` or an `OR` operator.
    - Edit the clause in the **Complete WHERE Clause** field as needed.

    Note:
    The **Add And** or **Add Or** buttons automatically generate the necessary syntax, UOM, and time zone conversions when possible.

11. Optional. Edit values for table parameters.

Table parameters apply only to linked tables. For more information, see Parameters for linked table queries.

12. Optional. For **Replacement Values**, choose attributes or literal values to return when the table query cannot find a matching row or encounters a null result.

Topics in this section

- Select first row matching criteria
- Summarize all rows matching criteria
- Table provided time series data
- WHERE clause syntax
- Parameters for linked table queries

Select first row matching criteria

The `Select first row matching criteria` rule enables you to specify the first row that matches the value returned from the `WHERE` clause.

Syntax

```
SELECT column FROM table WHERE where clause ORDER BY column ASC|DESC; options
and parameters
```

Arguments

- **`SELECT` *`column`***

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  In addition to `=`, `<>`, `>`, `>=`, `<`, `<=`, `LIKE`, and `IN` relational operators, you can specify `INTERPOLATE` to interpolate a value for the result column based on an interpolation of the specified input columns.

  Beginning with PI AF 2018, you can perform two kinds of interpolation:

  - Linear interpolation, in which interpolation is for 2 columns of data. To specify linear interpolation, use `INTERPOLATE(`*`column`*`,` *`value`*`)` syntax.
  - Standard bilinear interpolation, in which interpolation is for 3 columns of data. To specify bilinear interpolation, use `INTERPOLATE(`*`column_X`*`,` *`value_X`*`) AND INTERPOLATE(`*`column_Y`*`,` *`value_Y`*`)` syntax.

  Irregular bilinear interpolation and bilinear extrapolation are not currently supported.

  Note:
  The `INTERPOLATE` operator also returns values from outside the table range.

For more information on `WHERE` clause syntax, see WHERE clause syntax. For examples of `INTERPOLATE` operator usage, see Examples of linear and bilinear interpolation in table lookup data references.

- **`ORDER BY` *`column`***

  Optional. Specifies the sorting order so that the correct row is used when more than one row matches the `WHERE` clause. Ascending (**`ASC`**) order is the default unless descending (**`DESC`**) is specified.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Specifies the time zone of the source table. **Note:** OSIsoft recommends you set the time zone in the general description of the table, so that you do not need to specify it with each table lookup.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of `No Data` is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of `No Data` is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

Example

```
SELECT [Installation Date] FROM [Equipment Specifications] WHERE [Asset ID] =
'%Element%'
```

Examples of linear and bilinear interpolation in table lookup data references

Linear interpolation example

In a tank strapping table that contains a `Level` column and a `Volume` column, the following configuration string interpolates the volume based on the level reading:

```
SELECT Volume FROM MyTable WHERE INTERPOLATE(Level, @MyLevelReading)
```

Assume the sample table has the following rows:

Level	Volume
1	0.0
2	20.0
3	30.0
4	40.0
5	60.0
6	70.0

A `Level` reading of 2.2 results in a returned `Volume` of 22.0.

Bilinear interpolation example

In an air velocity table that contains an `X Horizontal Position` column, a `Y Vertical Position` column, and a `Velocity` column, the following configuration string interpolates the velocity based on the X horizontal and Y vertical positions.

```
SELECT Velocity FROM Table1 WHERE INTERPOLATE([X Horizontal Position]), @X) AND
INTERPOLATE([Y Vertical Position]), @Y)
```

Assume the sample table has the following rows:

X Horizontal Position	Y Vertical Position	Velocity
180	140	4.6
220	140	4.1
260	140	2.7
180	180	4.8
220	180	4.4
260	180	2.5
180	220	4.5
220	220	4.4
260	220	2.5

An `X Horizontal Position` of 245 and a `Y Vertical Position` of 165 results in a returned `Velocity` of 3.2171875.

Summarize all rows matching criteria

The `Summarize all rows matching criteria` rule enables you to perform a summary operation on the rows in a table column that match your selection criteria.

Syntax

```
SELECT summary(column) FROM table WHERE where clause; options and parameters
```

Arguments

- **`SELECT summary(column)`**

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets. You can select one of the following summary operations.

Operation	Description
Sum	The total of all row values.
Avg	The average of all row values.
Min	The minimum value of all rows.
Max	The maximum value of all rows.
Count	The number of rows.
StDev	The extent of deviation for all row values.
Var	The average measure of how far all row values differ from the mean.
None	When no operation is specified: ◦ If the result attribute is not an array, the value of the selected column in the first row that matches the `WHERE` clause is returned. ◦ If the result attribute is an array, an array with one value from each column of all rows that match the `WHERE` clause is returned.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  For more information on `WHERE` clause syntax, see WHERE clause syntax.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Specifies the time zone of the source table. **Note:** OSIsoft recommends you set the time zone in the general description of the table, so that you do not need to specify it with each table lookup.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of No Data is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of No Data is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

Table provided time series data

The `Table provided time series data` rule enables you to select a table containing values with a DateTime data type, and specify the column that contains time series data.

Syntax

```
SELECT column FROM table WHERE where clause; TC=column; options and parameters
```

Arguments

- **`SELECT` *`column`***

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  Optional. For more information on `WHERE` clause syntax, see WHERE clause syntax.

- **`TC` *`column`***

  Specifies the column that contains time series values. If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Not supported at the column level. You need to specify the time zone in the **Time Zone** field on the **General** tab for the overall table.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of No Data is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of No Data is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

WHERE clause syntax

The `WHERE` clause uses SQL syntax and conforms in general to the syntax described on the MSDN DataColumn.Expression Property (https://msdn.microsoft.com/en-us/library/system.data.datacolumn.expression(v=vs.110).aspx) page. You can type the clause directly into the **Complete WHERE Clause** field in the Table Lookup Data Reference window, or allow PSE to provide the correct syntax by using the **Column**, **Operator**, and **Attribute or Value** lists in the **Where** section. The `WHERE` clause is optional for the `Table provided time series data` rule.

The `WHERE` clause syntax follows these guidelines.

Column	Column names that contain non-alphanumeric symbols must be surrounded by brackets: `[Asset ID] ='%Element%'`
Operator	• The `INTERPOLATE` operator is available with the `Select first row matching criteria` rule. • The comparison operators `=`, `<>`, `>`, `>=`, `<`, `<=`, `IN`, and `LIKE` are supported for all lookup rules. You can use the * and % characters interchangeably for wildcard characters in a `LIKE` comparison. A wildcard is allowed at the start and end of a pattern, or at the end of a pattern, or at the start of a pattern, but not in the middle of a string. If the string in a `LIKE` clause contains a * or %, those characters should be enclosed in [ ] brackets.
Attribute or Value	*@attribute* Returns the value of the PI AF attribute. The attribute must be enclosed in [ ] brackets if it contains any non-alphanumeric character, including spaces, or includes a UOM or Time Zone specification. For example: `[height] >= @[Level Gauge;UOM=m]`
	literal • Strings should be enclosed in single quotes. • Numeric values are not quoted and should be in invariant format (where the decimal character is '.'). • Timestamps are best specified in `yyyy.mm.dd hh.mm.ss` format (0-23 for `hh`), and enclosed in the # character. For example: `#2015.01.30 14:00:00#`
	substitution Select or enter a substitution parameter in `'%substitution parameter%'` format. For more information on substitution parameters, see List of PI AF substitution parameters.

Parameters for linked table queries

The query for a linked table determines which data from an external source to include in the table. You can include parameters in the query, and, as you configure a table lookup data reference that uses the linked table, you can specify parameter values. This enables you to use a single linked table to obtain different results in every table lookup data reference that uses it.

Using parameters in a linked table query is useful, for example, to limit the number of rows returned from a very large external table. You can add conditions and parameters to return more targeted results, such as all rows that include a device or manufacturer ID number, specific for each table lookup data reference.

As you configure a linked table in the Table Link window, you can add table parameters to its query and set default values for them. Then, in the Table Lookup Data Reference window, as you define a data reference using the linked table, you can enter table parameter values specific for that data reference. You can also supply parameter values programmatically using AF SDK. Parameter values can be specific values or come from other attribute values or from pre-defined substitution variables, such as *%Element%*.

Add table parameters to a linked table query

As you define a linked table in the Table Link window, you can add parameters to the table query and set default values for them.

Procedure

1. Edit the text in the **Query** box to include the new parameter(s). Parameter names must begin with the @ character.
2. Click inside the **Parameters** table to display the new parameters from the query.
3. Enter default values for each parameter to determine the default results from the query and click **OK**. After you have added parameters to the query, you can specify values for them as you configure a table lookup data reference that uses the linked table.

Example

Consider the following query for a linked table named `MyTable`. The WHERE clause limits the selection from an external table (`BigTable`) to those rows with a particular `RowID`:

```
SELECT * FROM BigTable WHERE RowID = 101
```

Replace the fixed value 101 with a table query parameter `@id` (note that query parameter names must begin with the @ character):

```
SELECT * FROM BigTable WHERE RowID = @id
```

Now, for every table lookup data reference that uses `MyTable`, you can supply different table parameter values for `@id` to get different results from the query.

For example, in the Table Lookup Data Reference window, as you configure a data reference, enter `@AssetID` for the value of `@id` in the **Table Parameters** list. This sets `@id` to the current value of the attribute `AssetID`. The corresponding query for this would be:

```
SELECT Result FROM MyTable; @id=@AssetId
```

This query returns rows whose `RowID` matches the current value of `AssetID`.

URI Builder data references

In previous versions of PI Notifications, you could define a web link or a link to a client application such as PI WebParts or PI Vision as content within a notification. However, because the links were disconnected from the elements and attributes in PI System Explorer, only users

with PI Notifications installed were able to go find objects referenced in the email notification itself. Also, you could only include dynamic content as a value of a parameter.

With URI Builder in PI AF 2016, you can configure data references that create dynamic links on event frames, transfers, elements or models, and include the same attribute values and substitution parameters as other data references such as String Builder or Formula. URI Builder conforms to standard syntax rules (described in the Requests for Comments document RFC 3986 URI Generic Syntax (https://tools.ietf.org/html/rfc3986)) and automatically handles the escaping of characters which are complex and cumbersome to enter manually. It supports generic web links (that use http and https schemes), as well as PI Vision links.

You can also use PI Notifications 2016 to send the link to other users via electronic mail, an instant messaging client, or a client application such as PI Vision.

Create URI Builder data references

Copy an existing web link or PI Vision link and make it dynamic in a URI Builder data reference in which you substitute part of the URI with a value derived from an attribute, an attribute reference or a substitution parameter. You assign the dynamic web link value or PI Vision link value to an attribute template or attribute.

Before you start

When a URI Builder data reference is used in a template, value substitution takes place at run time.

Procedure

1. Open a web browser and locate the web link or the PI Vision link you use as a starting point in creating a link value on an attribute template or attribute.
2. Select the entire URI and copy it.
3. In the navigator pane of PI System Explorer, choose from the following actions.

To create a URI data reference on ...	Do this ...
A template	a. Click **Library**. b. Expand **Templates** in the browser tree. c. Choose a template type and select an existing template, or click **New Template**.
An element	a. Click **Elements**. b. Select an existing element in the browser tree, or click **New Element**.
A model	a. Click **Elements**. b. Select an existing model in the browser tree, or right-click and click **New Model**.
An event frame	a. Click **Event Frames**. b. Expand a search collection for event frames in the browser tree. c. Select an existing event frame, or click **New Event Frame**.

A transfer	a. Click **Event Frames**. b. Expand a search collection for transfers in the browser tree. c. Select an existing transfer or click **New Transfer**.

4. Choose from the following actions.
   - In the **Attribute Templates** tab, select an existing attribute template, or click **New Attribute Template** to create a new attribute template.
   - In the **Attributes** tab, select an existing attribute, or click **New Attribute** to create a new attribute.
5. In the attribute configuration panel, configure the attribute template or attribute as needed, and choose **URI Builder** from the **Data Reference** drop-down list.
6. Click **Settings**.
7. In the URI Builder Data Reference window, click in the **Paste a URI from a web browser to use as a template for configuration** field and paste the URI you copied in step 2.
8. Click **Continue**.

   The pasted URI is displayed in the URI Builder Data Reference window, broken down into its **Scheme**, **Address or Host**, **Port**, and **Path** field components. If the URI contains queries and fragments, those are also displayed in the **Query** and **Fragment** fields.
9. In the URI Builder Data Reference window, choose one of the following actions.

To configure ...	Do this ...
A dynamic web link	Modify the components of the pasted URI as needed. ◦ You can choose between **Scheme** values of http and https. ◦ You can change the **Port** value to any number between 1 and 65535. The default values for http and https schemes are 80 and 443 respectively. ◦ To make portions of the **Path** field dynamic, select a segment. Then click and select from the following options: ▪ A list of other attributes for the selected object. Note that attributes must appear in single quotes. ▪ A list of attribute references. You can also select **Search** to enter search criteria in the Attribute Search window. ▪ A list of commonly used substitution parameters. For more information, see List of PI AF substitution parameters. ◦ In the **Query** table, you can modify **Key** and **Value** settings as needed. ▪ To add a key, click . In the **Key** and **Value** fields, type a value or click and select from the same options as above. ▪ To delete a parameter, select a key or value and click .

A PI Vision link	a. Click the **PI Vision** option. b. In the **Display** field, modify the value as needed. You can also click ▸ and select from the following options: ▪ A list of other attributes for the selected object. Note that attributes must appear in single quotes. ▪ A list of attribute references. You can also select **Search** to enter search criteria in the Attribute Search window. ▪ A list of commonly used substitution parameters. For more information, see List of PI AF substitution parameters. c. In the **Parameters** table, modify **Key** and **Value** settings as needed. ▪ To add a key, click ✨. In the **Key** field, click ▸, select a parameter and assign a value. In addition to typing a value you can select from the same options as in step b above. ▪ To delete a parameter, select a key or value and click ✕. When linking to a display on an event frame, URI Builder automatically adds Asset, Start Time and End Time parameters. For more information on other parameters, see the PI Vision topic "URL parameters reference" in Live Library (https://livelibrary.osisoft.com).

10. Click **OK**.

Units of measure in PI AF

PI AF ships preloaded with numerous standard unit-of-measure classes and conversion factors. You can extend these classes by adding new units of measure, as well as new measurement classes. The implementation of the units of measure (UOM) feature in PI AF is based on the International System of Units (SI).

UOM database

All PI AF databases use the same set of UOMs, which are defined in the UOM database. All UOMs have the Read permission set, but other permissions can be set for the UOM database as a whole, as described in Configure security for the UOM database.

Beginning with PI AF 2017 R2, users with administration privileges can review audit trail data on the UOM database by selecting **File** > **Audit Trail Events** after they have clicked **Unit of Measure** in the navigator. For more information, see Track PI AF changes with Audit Trail.

UOM classes and canonical units

In PI AF, each unit of measure is based on a UOM *class*. Classes represent measurable properties, such as temperature, length, time, and mass. Each class has a *canonical unit*. This is the base unit from which PI AF converts values to other units when required. For example, the canonical unit for the `Length` class is `meter`, and the abbreviation is `m`.

UOM conversions in client applications

The UOM feature enables automatic unit conversions in client applications. For example, suppose a PI AF attribute has a UOM of `meter`. A PI ProcessBook user who is viewing that attribute value can choose to view the value in a different unit, such as `foot`. PI AF automatically converts the data from meters to feet.

UOM abbreviations

Prior to PI AF 2015 R2, all UOM abbreviations were case insensitive. A new option was added in PI AF 2015 R2 (v2.7.5) that allowed users to configure the PI AF server to support case-sensitive UOM abbreviations. Case-sensitive UOM abbreviations enable you to define abbreviations that differ only by case, such as `MV` (megavolt) and `mV` (millivolt), or `s` (second) and `S` (Siemens).

Note:
Beginning with PI AF 2017 R2, UOM case-sensitivity is enabled by default in a new install, but during an upgrade the current case-sensitivity setting remains unchanged. When case-sensitivity is enabled, only PI AF 2015 R2 or later clients can connect to that server.

Time integral UOMs

Beginning with PI AF 2017, you can view the UOMs associated with a time-weighted total (**Automatic** and **Per-Day**) of rate UOMs. Rate UOMs are from classes that contain time in the denominator of their **Base Units of Measure** (for example, Mass Flow Rate contains $\texttt{Time}^{-1}$, Pressure contains $\texttt{Time}^{-2}$). Each time-weighted total property displays the name of the UOM that will result from a summary Total (**Per-Day**) or TotalWithUOM (**Automatic**) data call.

UOM groups

Beginning with PI AF 2017 R2, UOM groups such as **Metric**, and **US Customary** are available so that you can map UOMs in a UOM class to the setting that is most appropriate for your location. By selecting the **Display UOM Group** option in **Tools** > **Options**, you can then see attribute values displayed in the preferred unit of measure instead of the value selected in an attribute template's **Default UOM** field. For more information on the US Customary UOM group, see the Wikipedia article United States customary units (https://en.wikipedia.org/wiki/United_States_customary_units).

Video

For information on how to edit units of measure, watch this video:

https://www.youtube.com/watch?v=YTG3Dk_SxNM&rel=0

Topics in this section

- Case sensitivity of UOM abbreviations
- Base classes and derived classes
- Create UOM classes
- Create units of measure
- Calculate conversion values for a UOM
- UOM groups
- Origin of UOMs

Case sensitivity of UOM abbreviations

A PI AF Client that is older than PI AF 2015 R2 (v2.7.5) cannot connect to a PI System that has been enabled to use case-sensitive UOM abbreviations. This is due to the possibility of UOM abbreviations being misinterpreted: they are often stored in data references, analysis rules, and client application files, such as spreadsheets and PI ProcessBook displays.

Note:
You must upgrade a PI AF Client to PI AF 2015 R2 or later before it can use case-sensitive UOMs.

References to UOM abbreviations

When UOM case-sensitivity is enabled in PI AF 2015 R2 or later, any stored reference to a UOM abbreviation requires that the correct case for the abbreviation be used. For example:

- Data references that are configured with UOMs (such as Formula data references) must use the correct abbreviation, or else the UOM lookup fails.
- Analysis rules, such as the Convert function, fail if configured with incorrect casing: `Convert('x', 'Kg')` instead of `Convert('x', 'kg')`.
- Applications or data references that programmatically access the UOM database must use the correct casing.

Note:
You cannot use the case-sensitivity option in the PI System to alert you to any configured uses that contain incorrect casing.

UOM names

Unlike abbreviations, UOM names remain case-insensitive in PI AF 2015 R2 or later when UOM case-sensitivity is enabled. However, when you create a UOM abbreviation, it cannot match the name of a different UOM, even if it differs in casing.

For example, the following is valid:

Name	Abbreviation
UOM1_name	UOM1
UOM2	uom1
UOM3	Uom1

The following is invalid and generates the error message: `'uom1' already exists in Unit of Measure 'UOM1' in Unit-of-Measure Database.`

Name	Abbreviation
UOM1	uom
UOM2	uom1

Configuration of case-sensitive UOM abbreviations

Although case-sensitive UOM abbreviations are enabled by default in new PI AF 2017 R2 installations, for older PI AF server versions you can change the setting for case-sensitive UOMs from the disabled state with the PI AF Diagnostics command-line utility (`afdiag`).

For more information on the `afdiag` utility, see PI AF Diagnostics utility.

Enable case-sensitive UOM abbreviations

To enable case-sensitive UOM abbreviations, you enter `afdiag /ucs` on the command line.

A new UOM, `millivolt` with a UOM abbreviation of `mV`, is added to the UOM database provided that there is no conflict with existing units of measure.

Disable case-sensitive UOM abbreviations

You can reset the PI AF server to case-insensitive UOM abbreviations with the same `afdiag` utility.

To disable case-sensitive UOM abbreviations, you enter `afdiag /ucs-` on the command line.

If any existing UOM abbreviations differ only by case, the utility does not allow the feature to be disabled and displays a list of the conflicting abbreviations. At a minimum, you need to remove the millivolt unit of measure that was created when you enabled case-sensitive UOM abbreviations, because `millivolt` (`mV`) conflicts with `megavolt` (`MV`).

Base classes and derived classes

PI AF has predefined a small set of *base classes*, and a larger set of classes that are derived from the base classes, called derived classes. Derived classes are simply classes that can be expressed in terms of the UOM base classes. For example, both the `Area` class and the `Volume` class are derived from the `Length` class. You can define virtually all units of measure in terms of a small group of UOM base classes.

PI AF uses the canonical unit for the base class to determine the canonical units for all classes derived from that base class. You cannot change the canonical unit for a base class or a derived class. For example, since the `Area` class is based on the `Length` class, the canonical unit for `Area` is the square meter (`m²`).

PI AF includes numerous standard UOMs, UOM classes, and conversion factors, but you can always add new ones. It is not typically necessary to create a new base class, although you might occasionally want to add one for non-physical measurements. For example, you could add a class called `Bytes` that could include UOMs of `bytes`, `KB`, `MB`, `GB`, and so on.

Topics in this section

- UOM base classes
- Common UOM derived classes

UOM base classes

The following predefined UOM base classes are available in PI AF.

Class	Canonical unit
Electric Current	ampere (A)
Length	meter (m)
Mass	kilogram (kg)
Moles (amount of substance)	mole (mol)
Plane Angle	radian (rad)
Quantity	count
Ratio	percent (%)
Temperature	kelvin (K)
Time	second (s)

Common UOM derived classes

The following predefined UOM derived classes are available in PI AF.

Class	Based on ...	Canonical unit
Angular Velocity	Plane Angle * $Time^{-1}$	radian per second (rad/s)
Area	$Length^{2}$	square meter (m^2)
Density	$Mass*Length^{-3}$	kilogram per cubic meter (kg/m^3)

Class	Based on ...	Canonical unit
Electric Charge	Electric Current * Time	coulomb (C)
Electric Potential	Electric Current^{-1} * Length^{2} * Mass * Time^{-3}	volt (V)
Energy	Length^{2} * Mass * Time^{-2}	joule (J)
Force	Mass * Length * Time^{-2}	newton (N)
Frequency	Time^{-1}	hertz (Hz)
Mass Flow Rate	Mass * Time^{-1}	kilogram per second (kg/s)
Power	Mass * Length^{2} * Time^{-3}	watt (W)
Pressure	Mass * Length^{-1} * Time^{-2}	pascal (Pa)
Specific Energy, Specific Enthalpy	Length^{2} * Time^{-2}	joule per kilogram (J/kg)
Speed	Length * Time^{-1}	meter per second (m/s)
Volume	Length^{3}	cubic meter (m^{3})
Volume Flow Rate	Length^{3} * Time^{-1}	cubic meter per second (m^{3}/s)

Create UOM classes

Create UOM classes when you need additions to the predefined classes in the UOM database.

Procedure

1. In the navigator, click **Unit of Measure**.
2. Choose from the following actions.
   - On the toolbar, click the **New Class** icon.
   - Right-click in the Unit of Measure browser and select **New Unit-of-Measure Class**.
3. In the Unit-of-Measure Class Properties window, fill out the properties on the **General** tab.
   a. In the **Name** field, enter a name for the class. See Valid characters in PI AF object names, if necessary.
   b. Optional. In the **Description** field, provide a description of the UOM class.
   c. In the **Canonical UOM** field, type the name of the canonical UOM for this class. If the UOM does not exist, PI AF creates it when it creates the new class.
   d. In the **Canonical UOM Abbreviation** field, enter a unique abbreviation.
   e. In the **Base Units of Measure** field, enter a calculation for the unit of measure. Choose from the following actions.

To ...	Do this ...
Add a base class calculation	a. Click . b. In the Add Base UOM Class window, select a base class from the **Base UOM Classes** list. c. In the **Base Power** field, type the exponent value or click one of the arrow keys until the exponent you want is displayed. d. Click **OK**. e. Repeat these steps as needed, to create a multiplication (positive base power) or division (negative base power) calculation.
Remove a base class calculation	a. Select a base class calculation entry. b. Click . c. In the Delete confirmation window, click **Yes**.

Note:
If you are defining a base class, leave the **Base Units of Measure** field blank.

4. Optional. You can add new units of measure to the class you are defining.
   a. Click the **Units of Measure** tab.
   b. Right-click in the list and select **New Unit of Measure**.
   c. In the Unit of Measure Properties window, define the new unit of measure. For more information, see Create units of measure.
5. Click **OK**.

Create units of measure

You create units of measure for a UOM class in the UOM database.

Procedure

1. In the navigator, click **Unit of Measure**.
2. In the Unit of Measure browser, select the class to which you want to add a new unit of measure.
3. Choose from the following actions.
   - On the toolbar, click **New UOM**.
   - Right-click the class in the browser and select **New Unit of Measure**.
   - Right-click in the UOM viewer and select **New Unit of Measure**.
4. In the Unit of Measure Properties window, enter a unique name for the unit of measure in the **Name** field.

   Note:
   The name cannot be the same as a previously defined UOM abbreviation or UOM name.
5. In the **Abbreviation** field, enter a unique abbreviation for the unit of measure.

If UOM case-sensitivity has been enabled, you can use the same abbreviation as a previously defined UOM abbreviation, so long as the case is different. For example, you can use `FR`, `fr`, `Fr`, and `fR` for four different UOM abbreviations.

> **Note:**
> The abbreviation cannot be the same as a previously defined UOM name, regardless of case.

6. Optional. In the **Description** field, enter a description for the unit of measure.
7. In the **Reference UOM** field, select the unit of reference class from which the new unit can be converted. The default value is the same as the read-only **Canonical UOM** value.
8. Choose one of the following conversion methods.

> **Caution:**
> Conversions that are based on **Formula** have some limitations as well as impact on performance. You should only select **Formula** when **Simple** conversions are inadequate.

Option	Description
Simple	Converts a UOM based on scaling factor and base offset values that you enter in the following fields. ◦ In the **Factor** field, enter the conversion factor from the reference UOM to the new UOM. For example, °C has a factor of 1 relative to kelvin. ◦ In the **Offset** field, enter a conversion offset from the reference UOM value. For example, °C has an offset of 273.15 from kelvin. **Note:** Although PI AF supports time-weighted totalizations that can automatically assign the appropriate units of measure, it is essential that the conversion factor of the UOM be defined to as precise a level as possible for this feature to activate. For example, if you were to define a flow rate UOM such as US gallons per day with a conversion factor of 4.38126E-08, automatic conversions for this UOM would be disabled, whereas if the conversion factor were 4.381263638889E-08, automatic conversions would be active.
Formula	Converts a UOM based on a complex conversion calculation. For more information, see UOM conversion calculation with a formula.

9. Optional. To establish an alternative UOM name so that users can see attribute values returned in their preferred UOM, in the **Mappings** list, select a UOM name from the **Mapping** list for each UOM group.

> **Note:**
> Although default mappings are provided for **Metric** and **US Customary**, you can modify them as needed. To modify multiple UOM mappings in UOM groups, OSIsoft recommends you use PI Builder.

10. Click **OK** to save the new UOM.

UOM conversion calculation with a formula

In general, OSIsoft recommends you specify UOM conversion calculations with a scaling factor and base offset from the Reference UOM value.

> **Caution:**
> Conversions that are based on **Formula** have some limitations as well as impact on performance. You should only select **Formula** when **Simple** conversions are inadequate.

Conversion calculation limitations

The following constraints exist with UOM conversion calculations.

Conversion type	Limitations
Factor	None.
Factor with offset	For delta conversion calculations, you need to define a separate UOM. See the Temperature (Delta) UOM class for examples.
Formula	• Delta calculations are not assigned UOMs. • Summary calculations over a range of data are not assigned UOMs. This affects analysis expression functions, such as Maximum, Minimum, Popular Standard Deviation, Range, Standard Deviation, and Total. It also affects programmatic Summary data calls against PI point data references with a UOM assigned. • Searches for attribute values exclude attributes whose value is measured in formula UOMs.

UOM formula evaluation

PI AF uses C# for UOM formula evaluation. Follow these guidelines:

- Write all Units of Measure in terms of the UOM abbreviation. If an abbreviation is not a valid C# variable name, enclose it in brackets.
- Adhere to C# evaluation rules.
- You can optionally invoke standard .NET static methods, such as **`Math.Log10()`**, to perform the computation. You are limited to what is available in the System Assembly (.NET Framework Math Class (https://msdn.microsoft.com/en-us/library/System.Math(v=vs.110).aspx)).

Formula conversion method example

As an example, `watts` (`W`) is the canonical UOM in the `Power` class. To create a UOM of `Decibel-milliwatts` (`dBm`) in the `Power` class, you could enter the following formula:

```
W = Math.Pow(10,(dBm - 30)/10)
```

```
dBm = 10 * Math.Log10( W ) + 30
```

Calculate conversion values for a UOM

Use the Unit of Measure Conversion Calculator to convert a given UOM quantity into an equivalent value for other members in its class.

Procedure

1. In the navigator, click **Unit of Measure**.
2. In the Unit of Measure browser, select a UOM class.
3. In the **Quantity** field of the **Conversion Calculator** pane, enter the quantity you want to convert.
4. Optional. In the **UOM** field, change the UOM from the default value to another UOM in the class.

   Converted values for each UOM in the class are displayed.

UOM groups

Beginning with PI AF 2017 R2, units of measure that are included as part of PI AF are mapped to UOM groups, such as **Metric** and **US Customary**. Once you have selected which UOM group you prefer to use at your location, you can display all attribute values with units of measure mapped to that UOM group rather than the default UOMs that are defined in attribute templates.

You select which UOM group you prefer in **Tools** > **Options**. When no preferred UOM group is selected, all attribute values are displayed with the default UOM defined in attribute templates, unless **Use Source Unit-of-Measure for attribute display** is also selected. For more information, see PI System Explorer customization options.

Sample mappings for a UOM class

As an example, the table below illustrates mappings for the Length UOM class.

UOM	Metric UOM group	US Customary UOM group
centimeter		inch
foot	meter	
inch	centimeter	
international nautical mile		
kilometer		mile
meter		yard
millimeter		sixteenth of an inch
sixteenth of an inch	millimeter	
yard	meter	

If a mapping is not defined, such as international nautical mile, a UOM is not mapped and uses the original UOM setting.

UOM database

UOM groups are available to all users and are stored on the PI AF server as part of the UOM database. Each user's UOM group selection in **Tools** > **Options** is also stored in the UOM database.

Users with permission to modify the UOM database can modify the UOM groups and mappings, including those shipped by default. For more information, see Configure security for the UOM database.

UOM database import/export

You can import and export UOM groups, as well as their mappings, with **File** > **Export to File** and **File** > **Import from File**. When you export, ensure that **Include All Referenced Objects** option is selected.

Manage UOM groups

In addition to the **Metric** and **US Customary** UOM groups that are installed with PI System Explorer, you can create additional UOM groups to which you can map units of measure. You can also modify and delete group names and descriptions.

Before you start

To create, modify, and delete UOM groups, you must have Write permission for the UOM database.

Procedure

1. To manage UOM groups, click **Unit of Measure** in the navigator.
2. Choose from the following actions.
   - On the toolbar, click the **UOM Groups** icon.
   - Click **File** > **UOM Groups**.
3. Choose from the following actions.

To...	Do this...
Create a new group	a. Click **New UOM Group**. b. In the Unit-of-Measure Properties window, enter a unique name for the UOM group in the **Name** field. c. Optional. In the **Description** field, enter a description for the UOM group. d. Click **OK**. The new group is added to the list in the UOM Groups window.
Modify a group	a. Select the group you want to modify and click **Properties**. b. In the Unit-of-Measure Properties window, make changes as needed to the name and/or description. c. Click **OK**.

Delete a group	a. Select the group you want to delete and click **Delete**. b. In the Delete confirmation window, click **Yes**. **Note:** If the deleted UOM group was previously selected in **Tools** > **Options** on the **General** tab, the **Display UOM Group** value is reset to **<None>**. Attributes are reset to the default UOM defined in the associated element template.

4. Check in your changes to the UOM database.

Origin of UOMs

Beginning with PI AF 2017 R2, you can check the origin of a UOM definition in the UOM database. The **Origin** field is included in the Unit of Measure Properties window, as well as the UOM viewer. It is also supported in PI Builder.

Possible values shown in the **Origin** column are:

Origin	Description
`Unknown`	The origin of how the UOM was defined is unknown.
`System Defined`	The UOM is system-defined and has not been modified. **Note:** If only a UOM mapping is changed, **Origin** remains set to `System Defined`.
`System Modified`	The UOM was system-defined but the **Name** and/or **Description** fields have been modified.
`System Replaced`	The UOM retains its original system-defined abbreviation and canonical UOM values but other fields besides **Name** and **Description** have been modified.
`User Defined`	The UOM was created by the user.

If a UOM is modified that results in **Origin** being set to either `System Modified` or `System Replaced`, but is later reset to the original definition, **Origin** also reverts to `System Defined`.

Security configuration in PI AF

PI AF 2015 (version 2.7) and later uses a security model that is similar to that which PI Data Archive uses. This model relies on integrated Windows security for authentication, but provides its own authorization to PI AF objects using PI AF identities and mappings.

> **Note:**
> The security configuration information described in this section presumes a PI AF 2015 server and a PI AF 2015 client.

The PI AF security model enables administrators to configure access for PI AF identities at each level of the PI AF hierarchy. PI AF uses Windows integrated security to grant or deny connection to the PI AF server, to view or edit databases, and to change collections.

For detailed information on how security is implemented on PI Data Archive, refer to the *PI Data Archive Security Configuration Guide* or see the Security configuration topic "Introduction to PI Data Archive security" in Live Library (https://livelibrary.osisoft.com).

Video

For information on how to create, map, and grant permissions to PI AF identities, watch this video:

https://www.youtube.com/watch?v=bWqHvoLuXzs&rel=0

Topics in this section

- PI AF identities and mappings
- Built-in PI AF identities
- Security hierarchy
- Security inheritance of PI AF objects
- PI AF access rights
- Deny permission option
- Security Configuration window
- PI AF server security
- PI AF database security
- PI AF collection security
- PI AF object security
- Configure security for the UOM database
- Differences from PI Data Archive security model

PI AF identities and mappings

A PI AF identity represents a set of access permissions on the PI AF server. Each PI AF mapping points from a Windows user or group to a PI AF identity. Beginning with PI AF Server 2015 (v2.7), you cannot directly grant a Windows user or group access to a PI AF server resource (such as an element collection or objects). Instead, you create a PI AF identity that has that

access and then you create a PI AF mapping between the Windows user or group and that PI AF identity.

Members of the Windows groups that are mapped to a PI AF identity are automatically granted the access permissions for that PI AF identity. For example, in the following illustration, the PI AF identity called `Engineers` has read/write access to the Elements collection. Because the Active Directory (AD) group `Engineering Team` is mapped to `Engineers`, all the members in that AD group get read/write permission for the Elements collection.

AD group mapping to a PI AF identity

Multiple identities

A single Windows user can be mapped to multiple PI AF identities, typically via mappings of the various Windows group memberships to which he or she belongs. A user is granted permissions based on *all* the PI AF identities to which he or she is mapped. Effective permissions are determined by taking the union of all identities' allowed permissions and removing the union of all denied permissions. For example, in the following illustration, the Windows user Bob belongs to both AD groups. Bob therefore gets the permissions that are configured for `PI AF IDENTITY`$_1$ and `PI AF IDENTITY`$_2$.

Windows user with cumulative access permissions

Additionally, a user must have read permission on a PI AF database to be able to read any object within it. Likewise, a user must have write permission on a PI AF database to write to any object within it.

For more information on working with identities and mappings, see Manage identities and Manage mappings.

Built-in PI AF identities

PI AF includes the following built-in PI AF identities:

PI AF identity	Description
`Administrators`	By default, this identity has all access permissions to every collection and object on the PI AF server, including all databases. It cannot be modified or deleted. Mappings, however, can be added and removed, and this identity can be denied access permissions to objects if the need arises. OSIsoft recommends that access to this identity be restricted to only a few users.
`Asset Analytics`	(Part of PI Analysis Service installation.) This identity has the necessary access permission to work with analyses. By default, the account used to run PI Analysis Service is mapped to this identity during installation. Mappings to this account can be added or removed.
`Asset Analytics Recalculation`	(Part of PI Analysis Service installation.) This identity has Execute permission, allowing users mapped to it to backfill and recalculate analyses.[1]

PI AF identity	Description
`Engineers`	This identity has the same privileges as `Administrators`, with the exception of the **Admin** (a) permission. This identity is also not allowed to delete PI AF databases. OSIsoft recommends that this identity be restricted to those users who are defining the asset database. Additional identities should be created to narrow the scope of access within PI AF.
`Notifications`	(Part of PI Notifications Service installation.) This identity has the necessary access permission to work with notification rules. By default, the account used to run PI Notifications Service is mapped to this identity during installation. Mappings to this account can be added or removed.
`Owner`	This read-only identity can be explicitly added to the security configuration of specific PI AF objects to enable administrator users to configure privileges for the owner of an object. The following restrictions apply: • You cannot add mappings to this identity. • You cannot modify, disable, or delete this identity. • You cannot remove or deny Read permission for this identity. The owner of an object always has Read and Read Data permissions on that object.
`World`	This identity has read access permissions to every collection and object on the PI AF server. It cannot be modified or deleted. Mappings, however, can be added and removed. By default, this identity is mapped to the Windows `Everyone` users group.

[1] If you have installed PI AF 2017 (version 2.9) or later versions for the first time, users need to be added to the `Asset Analytics Recalculation` identity in order to backfill or recalculate. If you are upgrading from versions prior to PI AF 2017 (version 2.9), all users will automatically be mapped to `Asset Analytics Recalculation` identity. It is recommended that upon upgrading, such automatic mapping for all users is removed and users that require backfilling or recalculation permissions are explicitly mapped.

Security hierarchy

PI AF uses Windows integrated security to authenticate users and establish their PI AF identities through mappings. PI AF uses the PI AF identities to control read, write, delete, and various other permissions on PI AF components shown in the following illustration. Each securable PI AF object (element, event frame, and notification, and so on) throughout the hierarchy has an associated security descriptor that contains the access permissions information for that object.

All PI AF objects of the same type belong to a collection. For example, every PI AF element in a database belongs to the Elements collection for that database. Each collection also has an associated security descriptor that contains access permission information. Security descriptors for some collections are configured for an entire PI AF server (such as Identities and Mappings), whereas others (such as Analyses, Elements, and Event Frames) can be configured for a specific database.

PI AF hierarchy of securable collections

For more information on collection security, see PI AF collection security.

Security inheritance of PI AF objects

The following table provides details on the security that is required to create specific PI AF objects. Note that the Owner of an object always has Read and Read Data permissions on that object.

Object	Security inheritance
Element	A child element inherits the security of its parent element if it is added as a Parent-Child (strong) or Composition reference. For more information, see Permission inheritance of element objects. The security rights of an element created at the PI AF SQL Server database level are derived from the Element collection associated with the PI AF SQL Server database. For more information, see PI AF collection security.
Event frame	A child event frame inherits the security of its parent event frame if it is added as a Parent-Child (strong) reference. If an event frame does not have a Parent-Child reference, security rights are derived from the element template if it was created from a template. If it was not created from a template, security rights are derived from the Event Frame collection associated with the PI AF SQL Server database.
Transfer	If a transfer is created from a template, security rights are derived from the template. If it was not created from a template, security rights are derived from the Transfer collection associated with the PI AF SQL Server database.
Case	Security rights are derived from the analysis that owns the case.

For all other types of objects, initial security access rights are calculated from the corresponding collection associated with the PI AF SQL Server database. For example, security access rights for a notification are initially derived from the PI AF SQL Server database's Notification Rule collection.

Note:
When an analysis or analysis template is associated with a legacy notification or notification template, (Notifications 2012 or earlier), security access rights for the two objects are synchronized.

Permission inheritance of element objects

When you change access permissions for an element, the access permissions for any parent or child elements might also change. The behavior depends on the reference type.

Parent-child reference type

When an object or collection is created, a default set of access permissions is assigned, based on the access permissions that are set on the parent. When access permissions are set on a parent, the following **Child Permission** settings in the Security Configuration window are evaluated:

Option	Description
Do not modify child permissions	Prevents access permissions that have been set for the current object or collection from being replicated to child collections and objects in the PI AF hierarchy. This option is the default when the connected PI AF server is running 2.5 and earlier versions.

Option	Description
Update child permissions for modified identities	For each selected item on the **Items to Configure** list in the Security Configuration window, replicates the access permissions for all child collections and objects for each identity on the **Identities** list whose access permissions have been modified. This option is the default when the connected PI AF server is running 2.6 and later versions. This option is not available when the connected PI AF server is running 2.5 and earlier versions.
Replace child permissions for all identities	For each selected item on the **Items to Configure** list in the Security Configuration window, replaces all child permissions for every identity on the **Identities** list with the parent access permissions. **Tip:** Before you apply this option, OSIsoft recommends that you review access permission settings for all items on the **Items to Configure** list to avoid unintentionally overwriting custom permissions that may have been applied elsewhere in the collection hierarchy. Review the following example for clarification.

Examples

The following hierarchy has three elements, each with a different access permissions setting.

Sample element hierarchy

- The `Administrators` and `World` identities have access permissions to all three elements: `EasternUS`, `SavannahSite`, and `ProductionLine1`.
- The `Savannah_IT` identity has access to the `SavannahSite` element.
- The `SavnEngineers` identity has access to the `ProductionLine1` element.

Access permissions for sample hierarchy

Suppose you want to add a `CorporateEngineering` identity to each element in the hierarchy. You would add the identity to the parent element `EasternUS`:

Add identity to parent element

To replicate the parent permissions without affecting identities already assigned access permissions, you would set the **Child Permissions** option to **Update child permissions for modified entries**. The security string for all three elements shows that the `CorporateEngineering` identity has been added, but the other identities remain unchanged:

Replicate identity and modify existing access permissions

To replicate the parent permissions for all identities down the hierarchy, you would set the **Child Permissions** option to **Replace child permissions for all identities**. The security string for all three elements shows that the `CorporateEngineering` identity has been added, but has replaced the access permissions with those assigned to the `EasternUS` element:

Replicate identity and replace existing access permissions

Notice that the `Savannah_IT` and `SavnEngineers` identities no longer appear in the security string for the `SavannahSite` and `ProductionLine1` elements because they were not included in the **Identities** list of the parent `EasternUS` element.

Composition reference type

Access permissions for child and parent are always the same.

If you change the access permissions for the child, the parent access permissions are automatically changed to match the child permissions. Similarly, if you change the access permissions for the parent, the child access permissions are automatically changed to match the parent permissions. These changes cascade down (and up) through the hierarchy.

Weak reference type

Access permissions are never inherited.

PI AF access rights

The following table describes the access permissions you can assign to PI AF identities for all objects in the PI AF hierarchy.

Access right	Security string abbreviation	Definition
`Read`	`r`	Enables a user to view the object. `Read` security rights are required to view the object in client applications. The following objects always have `Read` permission regardless of their security settings, so long as the logged-in user has Read access to the PI System and the PI AF SQL Server database. The one exception is the notification contact template, where the logged-in user only requires Read access to the PI System: • Analysis template • Categories • Element template • Enumeration set • Event frame template • Model template • Notification contact template • Notification template • Reference type • Transfer template • UOM database
`Write`	`w`	Enables a user to create and modify an object. The exception is that event frames and transfers also require `Write Data` permission on the element template from which they are created, and cases require `Write Data` permission on the analysis in which they are contained.
`Read/Write`		Enables a user to read and write to the associated object. When selected, automatically selects the `Read` and `Write` permissions.
`Read Data`	`rd`	Enables a user to read non-configuration values from attributes of elements (the **Configuration Item** property for an attribute is cleared). Additionally, this permission controls whether a user can see transfers created from a specific transfer element template. Similarly, it controls whether a user can see cases created in a specific analysis. If the following objects have `Read Data` permission, they are also granted `Read` permission: • Case • Element • Event frame • Model • Notification • Transfer
`Write Data`	`wd`	Enables a user to write non-configuration values to attributes of elements (the **Configuration Item** property for an attribute is cleared). Additionally, this permission controls whether a user can create or modify event frames or transfers created from a specific transfer element template. Similarly, it controls whether a user can create or modify cases in a specific analysis.
`Read/Write Data`		Enables a user to read data and write data to the associated object. When selected, automatically selects the `Read` and `Write Data` permissions.

Access right	Security string abbreviation	Definition
`Delete`	`d`	Enables a user to delete an object. Delete security rights are required to delete an object, either directly or indirectly by removing it from other objects.
`Execute`	`x`	Enables a user to queue backfilling or recalculation of analyses in the analysis service. It also enables a user to perform most actions on an analysis case.
`Admin`	`a`	Enables a user to modify the security settings, or owner, of an object. Administration security rights are required to force an **Undo Check Out** on an object that is checked out to another user, as well as to lock and unlock an event frame. **Note:** Users with the administration permission on the PI AF server object are granted all rights not only to the system, but to all objects within the system, including databases.
`Subscribe`	`s`	Enables a user to subscribe and unsubscribe to a notification.
`Subscribe Others`	`so`	Enables a user to subscribe and unsubscribe other users to a notification.
`Annotate`	`an`	Enables a user to annotate and acknowledge event frames. **Note:** This access right was added in PI AF 2016. After an upgrade from earlier server versions, objects with the `Write Data` (`wd`) access right are granted the `Annotate` access right automatically. Both client and server upgrades must use this new permission.

PI Data Archive permissions

In order to connect to a PI Data Archive server, PI AF requires that `Read` permission be configured on that server. Note that because determining the permissions on a PI Data Archive server requires an actual connection to be made, the full list of PI Data Archive servers configured on a client machine is always available. PI Data Archive also always has `Delete` permission.

Deny permission option

You select the **Deny** option for these cases:

- To exclude a subset of an identity mapping that has allowed permissions.
- To exclude one special permission when you have already granted full control to an identity.

Note:
PI Module Database does not support the Deny option. If you are using both PI MDB and PI AF, avoid the Deny option to prevent synchronization problems.

Security Configuration window

You can open the Security Configuration window anywhere in PI System Explorer where a **Security** link is displayed in a window or **Security** is displayed on a context menu.

Security Configuration

Items to Configure list

The content of the **Items to Configure** list is determined by the hierarchy context and indicates whether you can configure access permissions for the entire PI AF hierarchy, a single database and containers, or a single container and single object.

To configure access permissions for only some of the items, you can clear those you do not want to configure.

Identities list

The **Identities** list contains a list of identities that have permissions for *all* checked items in the **Items to configure** list. You use the **Add** and **Remove** functions to manage which identities appear on the list.

If the currently connected PI AF server does not support identities (version 2.6 or earlier), the Security Configuration window displays access permissions for Groups or Users and you can add and remove Windows accounts using standard Active Directory windows.

Permissions list

As each identity is selected in the **Identities** list, the permissions associated with that identity for the checked entries in the **Items to Configure** list are shown in the **Permissions** list. You can allow or deny permissions as desired for each identity without losing the changes. The access permissions you set for each identity are retained until you click the **OK**, **Cancel**, or **Apply** button.

Note:
If more than one **Items to Configure** entry is selected and the currently selected identity has one of the permissions in one or more, but not *all* entries selected in the **Items** list, the checkbox for that permission displays a dot (■) to indicate some entries in the **Items** list have the permission set whereas some do not.

PI AF server security

When PI AF Server is first installed, admin access (all permissions) is given to the built-in `Administrators` identity, and read access to all objects and collections is given to the `World` identity. For all other identities that are mapped to Windows users and groups, an administrator needs to configure appropriate access to the PI AF server, databases, collections and objects.

Configure security for a PI AF server

Configure access permissions for a connected PI AF server so that a PI AF identity can access databases, collections and objects.

Before you start

To connect to a PI AF server, an identity must have read permissions to the PI AF server object.

Procedure

1. Choose one of the following methods to open the Security Configuration window for a PI AF server:

To open from ...	Do this ...
Servers window	a. Select **File** > **Connections**. b. In the Servers window, right-click the default connected ( ) PI AF server and click **Security**.
PI AF Server Properties window	a. On the toolbar, click . b. In the PI AF Server Properties window, click the **Security** link.
Select Database window	a. On the toolbar, click the **Database** button. b. In the Select Database window, click the **Edit Security** button.

   In the **Items to Configure** list of the Security Configuration window, the server and every collection is selected.

2. Clear those collections that you do not want to be assigned the same permission settings as the server. For example, you might want to assign access rights for server-wide collections such as Mappings and Identities, but use a different set of access permissions for databases and their collections.

3. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set.

Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

4. Select one of the **Child Permissions** options. For more information, see Permission inheritance of element objects.
5. Click **OK**.

PI AF database security

To view a database, an identity must have at least read permissions to the database object. You configure permissions individually for each database, or for the entire databases collection.

Beginning with PI AF 2017 R2, it is no longer necessary for users to have write permission on a PI AF database in order to have write permission on other objects in the database. Users need only be assigned an identity with write access to objects such as elements to be able to edit them. To prevent users from having write access to specific objects, assign them to an identity with read-only access to those objects.

Topics in this section

- Configure security for new PI AF databases
- Configure security for a single PI AF database

Configure security for new PI AF databases

Configure default PI AF access rights for all identities that need access to databases on a PI AF server.

Procedure

1. On the toolbar, click the **Database** button.
2. In the Select Database window, click the **Edit Security** button.

   In the **Items to Configure** list of the Security Configuration window, the server and every collection is selected.
3. In the **Items to Configure** list, clear the server and the following collections: **Contacts**, **Notification Contact Templates**, **Identities**, and **Mappings**. **Databases** and all **Database - *Collections*** should be selected.

4. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check box beside each permission you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are checked). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

5. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
6. Click **OK**.

Configure security for a single PI AF database

For a single database, configure PI AF access rights for identities that need access to its collections and objects.

Procedure

1. On the toolbar, click the **Database** button.
2. In the Select Database window, right-click a database in the **Databases** list and click **Security**.

   In the **Items to Configure** list of the Security Configuration window, the database and every collection in the database is selected.
3. Optional. Clear any collection that you do not wish to modify from their default settings and assigned identities.
4. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are checked). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see PI AF identities and mappings.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

5. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
6. Click **OK**.

PI AF collection security

The security descriptor for a PI AF collection determines access permissions to that collection, as well as default access permissions for new objects in the collection. For example, the Elements collection permissions determine which identities can create and delete elements. The Elements collection permissions are replicated as the security descriptor for any newly-created Element objects.

You can configure access permissions to collections at several points in the PI AF hierarchy. You can set them at the server level so that permissions assigned to identities on the server are also assigned to the same identities for every collection in every database (see Configure security for a PI AF server). You can set them at the database level so that permissions assigned to identities in a database are also assigned to the same identities for every collection in that database (see Configure security for a single PI AF database).

Note:
Library objects (Templates, Enumeration Sets, Reference Types, Tables, Table Connections, and Categories) always have Read (r) permission regardless of their security settings.

Configure security for collections

Assign access permissions for a specific collection in a PI AF database.

Procedure

1. Choose one of the following methods to open the Security Configuration window for a database collection:

To open ...	Do this ...
Elements collection	In the Elements browser, right-click the Elements collection ( ) and click **Security**.
Event Frame collection	In the Event Frames browser, right-click an event frame search ( ) and click **Security**.
Transfer collection	In the Event Frames browser, right-click a transfer search ( ) and click **Security**.
Element Templates collection	In the Library browser, right-click **Element Templates** and click **Security**. **Note:** Access permissions for the Element Templates collection also set access permissions for Event Frame Templates, Model Templates, and Transfer Templates.
Enumeration Sets collection	In the Library browser, right-click **Enumeration Sets** and click **Security**.
Reference Types collection	In the Library browser, right-click **Reference Types** and click **Security**.
Tables collection	In the Library browser, right-click **Tables** and click **Security**.
Table Connections collection	In the Library browser, right-click **Table Connections** and click **Security**.
Categories collection	In the Library browser, right-click **Categories** and click **Security**.

In the **Items to Configure** list of the Security Configuration window, the collection is selected.

2. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check box beside each permission you wish to set. c. Repeat for each identity.

Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

PI AF object security

You can set specific access permissions for an identity that differ from the default settings inherited from elsewhere in the PI AF hierarchy on any object (or object group) and collection in a database.

Topics in this section

- Configure security for objects
- Configure security for analyses and analysis templates

Configure security for objects

Set access permissions for identities to objects in the browser or the viewer in situations where access needs to be different from inherited permissions. You can also set custom permissions for identities to a specific collection in the Library.

Procedure

1. Choose from the following actions:

To set access permissions for ...	Do this ...
A browser object	Right-click the object and click **Security**.
Multiple objects	a. With multiple objects listed in the viewer (after a search, for example), use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL +<click>) objects. b. Right-click and click **Security**.

2. In the **Identities** list, review the identities and validate their permissions. You can also add to and remove an identity from the list.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

Configure security for analyses and analysis templates

Use the Security Configuration window to manage access permissions for PI AF identities that can work with analyses or analysis templates.

For information on specific permissions that are needed to work with analyses, see PI AF access rights.

Procedure

1. Choose from the following actions:

To ...	Do this ...
Configure access permissions for an analysis	a. In the Elements browser, select the element that contains the analysis. b. Click the **Analyses** tab. c. Right-click the analysis name and click **Security**.
Configure access permissions for an analysis template	a. In the Library browser, select the element template that contains the analysis template. b. Click the **Analysis Templates** tab. c. Right-click the analysis name and click **Security**.

Configure access permissions for all analyses in a PI AF database	a. On the toolbar, click **Database**. b. In the **Select Database** window, right-click the database name and click **Security**. c. In the Security Configuration window, clear the **Item** check box above the **Items to Configure** list. d. Select the ***database* - Analyses** item.
Configure access permissions for all analysis templates in a PI AF database	a. On the toolbar, click **Database**. b. In the **Select Database** window, right-click the database name and click **Security**. c. In the Security Configuration window, clear the **Item** check box above the **Items to Configure** list. d. Select the ***database* - Analysis Templates** item.

2. In the **Identities** list, review the identities and validate their permissions. You can also add to and remove an identity from the list, as described in Configure security for objects.
3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

Configure security for the UOM database

You cannot set permissions for individual UOMs or UOM classes. However, you can set permissions for the entire UOM database.

Before you start

UOMs always have the Read permission regardless of other security settings.

Procedure

1. In the navigator, select **Unit of Measure**.
2. On the toolbar, click **UOM Security**.

   In the **Items to Configure** list of the Security Configuration window, the **Unit-of-Measure Database** item is selected.
3. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as described in Configure security for objects.
4. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
5. Click **OK**.

Differences from PI Data Archive security model

Although the security model is similar to that which PI Data Archive uses, there are a number of differences:

- The Deny privilege is supported. See Deny permission option.
- A more expansive set of access permissions is available. See PI AF access rights.
- The equivalent of `PI user` and `PI group` identities are not supported.
- PI trusts and PI explicit logins are not supported.
- The concept of an undeletable flag on an identity or mapping is not supported.
- Mappings cannot be disabled. Only PI AF identities can be disabled.
- A single Windows user identity can be mapped to more than one PI AF identity.
- A different built-in set of identities is installed: `Administrators`, `Engineers`, and `World`.

Event frames in PI AF

Capturing important events in your process and collecting relevant data around those events can help analyze why they occurred. For example, you can closely monitor events such as asset downtime, process excursions, equipment startup or shutdown, and environmental excursions to identify possible causes or potential points of failure.

Collecting data around repeatable time periods such as product tracking batches, product runs, or operator shifts can help make those processes more efficient. The capture of comprehensive data associated with such an event helps track, compare, or analyse the process or event.

Just as elements allow you to collect and store data about assets, event frames allow you to collect and store data about events. OSIsoft recommends you use asset analytics to track your events using event frames. PI Datalink, PI Vision, and PI WebParts are the client tools that support event frame visualization.

An event frame encapsulates the time period of the event with relevant, comprehensive asset data:

Sample event data

Name	Start Time	End Time	Primary Element	| Operator	| Phase	| Reason Code	| Level.Start	| Temp.Max	| Temp.Start
Template: Downtime									
DT20120805 22:56	8/5/2012 10:56:00 PM	8/5/2012 11:00:00 PM	RE200	Vivian White	Material Add 1	DRIVE FAILURE	0 L	12.935262298...	0.41429677605629 deg C
DT20120805 23:00	8/5/2012 11:00:00 PM	8/6/2012 12:00:00 AM	RE400	Don Wallace	XFER_OUT	No Oil	69.978393554...	110.66223907...	40.0944023132324 deg C
DT20120805 23:26	8/5/2012 11:26:00 PM	8/5/2012 11:29:00 PM	RE100	Bobby Johnson	Material Add 1	No Oil	4.6750192642...	5.4764796495...	3.44453287124634 deg C
DT20120805 23:56	8/5/2012 11:56:00 PM	8/6/2012 12:00:00 AM	RE200	Vivian White	Inactive	DRIVE FAILURE	3.5982928276...	8.9376313209...	8.93763160705566 deg C
DT20120806 0:14	8/6/2012 12:14:00 AM	8/6/2012 12:18:00 AM	RE400	Don Wallace	XFER_OUT	ESTOP	74.283164978...	56.506531524...	56.5065307617188 deg C
DT20120806 0:26	8/6/2012 12:26:00 AM	8/6/2012 12:29:00 AM	RE100	Bobby Johnson	Dwell	UNKNOWN	62.131874084...	101.77695465...	98.174674987793 deg C
DT20120806 0:56	8/6/2012 12:56:00 AM	8/6/2012 1:00:00 AM	RE200	Vivian White	Agitate	ESTOP	46.672977447...	62.632131067...	62.6321296691895 deg C
DT20120806 1:14	8/6/2012 1:14:00 AM	8/6/2012 1:18:00 AM	RE400	Don Wallace	Heat	DRIVE FAILURE	68.781196594...	96.730087280...	72.2844924926758 deg C

Videos

For information on event frames, watch this video:

https://www.youtube.com/watch?v=4XPL7J7u61U&rel=0

Alternatively, you can view all the videos on the event frames playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2JbcunItGs7JS6kFl3WYMb-p3D&rel=0

Topics in this section

- Structure of event frames
- Advantages of event frames
- Examples of event frames
- Event Frames or Batch?
- Ways to create event frames
- Visualization of event frames
- Event frame templates

- Create event frames
- Value capture for event frames and transfers
- Lock event frames or transfers
- Acknowledgement of event frames
- Transfers

Structure of event frames

Each event frame has a name, start time, end time, one or more attributes, and one or more referenced PI AF elements. As with elements, you should create event frame templates to standardize and manage the attributes for different types of events.

With event frames you can easily search the PI System for the events themselves, rather than search for events by time. You can configure event frames to return all related event data automatically in real-time so that you do not need to query multiple systems for event and process data, and then merge them together manually. You can also set up event frames to retrieve historical data.

When to use event frames

There are two categories of trackable events that would fit an event frame profile:

- "Good" events: Events that you want to track as a normal part of business, such as product tracking, shifts, and so on.
- "Bad" events: Events that are unexpected and need to be analyzed and perhaps fixed quickly if they ever occur, such as expected shutdowns or excursions. These are events that you want to track and report in aggregate, over a period of time.

Asking questions such as these can help identify events or conditions that must be tracked:

- What are all the times that event X occurred on this type of asset?
- Can I associate data from different tags for a time-range, or for a single point in time?
- What is the associated data for a particular time period when a problem occurred or may occur in the future?
- What are the critical process events that someone needs to be notified on?
- Are there digital states for PI system tags that are significant when they change, and must they trigger other actions?

Advantages of event frames

Event frames provide these advantages:

- **Flexibility**

  Event frames:

  - Reference multiple elements within the same event.
  - Support multiple overlapping events on a PI AF element.
  - Capture any event; a "batch" is just one type of capturable event.

- **Easy search options**

  Searches can be specified on a number of configurable event frame attributes: it is optional to specify a time range. For example, you can search just by entering the name of the event frame. The most commonly searched event frame attributes can be configured as indexed attributes through event frame templates, which speeds up end-user searches.

- **Scalability**

  Event frames are extremely scalable whereas search performance degrades with a large number (tens of thousands) of batches.

Examples of event frames

Example of event frames in a wind power generation company

Take the example of a wind power generation company that has different types of windmills across different locations and uses PI AF to organize their data. Their asset framework structure is based off a base element template of type "windmill", and has various child templates based off OEM, model, and megawatt ratings. The wind operator may want to win favorable warranty contracts by showing that the blades are operating safely. Or, the operator may need to monitor parameters such as oil bearing pressure, maximum voltage, power-factor, or performance of electronic and mechanical brakes in the minutes before process trips occur. In all these cases, you can associate event frames with the specific event frame attributes.

The sample windmill element may include these asset attributes and associated event frame attributes:

Asset attribute	Event frame attribute
RPM	Instantaneous and maximum speeds
Voltage	Maximum voltage
Power	Power at current time
Yaw	Yaw at current time
Pitch	Pitch at current time
Oil bearing pressure	Maximum oil bearing pressure

Since all types of turbines share many identical attributes, you can create just one event frame template and use it to monitor similar events across the different assets. For example, you may be interested in the RPM attribute to capture a speed-based event. Using the special notation `.\Elements[.]|RPM` in the template enables you to use the template on any windmill, and access the attribute of the particular referenced element.

Example of event frames using Asset Analytics

For an example of creating event frames using asset analytics, see Create event frames automatically to track inefficiency.

Event Frames or Batch?

Batch provides you with a means to generate batch event data based on PI tags. Event Frames provide you with a way to track and analyse process and business data related to events that are defined on PI AF attributes.

Different companies and users come from different levels of PI System implementation and use of PI System features; hence, the recommendation of whether to choose Batch or Event Frames is highly subjective. You may choose Batch or Event Frames depending on your process and business needs, such as:

- You are a Batch user and use RtReports.

  You must continue to use Batch. Typical Batch use-cases include paper grade runs, chemical batch processing operations, and so on.

- You do not currently use Batch but have batch use-cases, or must use RtReports.

  You must use Batch. OSIsoft will provide a migration path in the future when the PI System begins to support these functions.

- You do not use Batch and do not have batch use-cases.

  You must use Event Frames. Typical Event Frame use-cases include downtime or outages in power industries, excursions in water utility industries, and so on.

> **Note:**
> OSIsoft does not recommend that you run dual or parallel instances of Batch and Event Frame interfaces in production as it adds to your migration complexity.

Unsupported PI System functionality with Event Frames

- RtReports - PI AF and Event Frames are not currently supported by RtReports; however, you can use PI System access products like PI OLEDB Enterprise to write SQL queries for third-party reporting solutions.

Ways to create event frames

Although you can create event frames with several different OSIsoft products, OSIsoft strongly recommends that you use asset analytics.

Asset analytics

PI Data Archive supports creation of single-layer events and can trigger expressions that open or close event frames. See Event frame generation analyses.

PI Event Frame Generator (PI EFGen)

PI EFGen uses both PI SDK and AF SDK to create a hierarchy of events or to convert a PI BaGen structure to an event frame generator structure. Use PI System Explorer or PI Builder to create the event frame templates, associated attributes and PI point references.

For instructions on how to create event frames with PI EFGen, watch this video:

https://www.youtube.com/watch?v=UbideDwhawg&rel=0

PI interfaces for batch and manufacturing execution systems

PI Batch Framework 3.x and later versions can create event frames within a PI AF database or PI Batch objects within a PI Batch database. For more information on creating event frames and using PI interfaces to populate the PI AF database with events-based data, see:

- "PI interfaces for batch and manufacturing execution systems" in Live Library (https://livelibrary.osisoft.com)
- "PI Event Frames Interface Manager" in Live Library (https://livelibrary.osisoft.com)

Programmable interfaces

You can create your own custom program using AF SDK and PI ACE to create and monitor events.

Manual creation of event frames

OSIsoft strongly discourages manual creation and management of event frames and encourages you to use PI Data Archive with asset analytics support or PI Batch, depending on your process needs. See Event Frames or Batch?

Visualization of event frames

You can use PI Datalink, PI Vision, and PI WebParts client tools to visualize event frames. You can also use PI OLEDB Enterprise, PI JDBC Driver, or PI Web Services to integrate event frame data into other third-party reporting tools.

> **Note:**
> PI ProcessBook, PI BatchView, and PI Manual Logger do not currently support event frame visualization.

- **PI Datalink**

  PI Datalink support for event frames includes exploring and comparing hierarchical events. For more information, see "Events in worksheets" in Live Library (https://livelibrary.osisoft.com).

- **PI Vision**

  PI Vision enables you to view and analyze your PI System data during the time range of a particular event. For example, you may want to examine the performance of an asset during an operator shift or compare the data for several assets during a downtime period.
  PI Vision supports event frames through the "Related Events" viewer. For more information, see "Analyzing and comparing events" in Live Library (https://livelibrary.osisoft.com).

- **PI WebParts**

  PI WebParts does not include specific features related to event frame visualization. However, you can create a data set based on a relational data source that retrieves event frame data via PI OLEDB Enterprise. This event frame data can then be displayed in a PI Table web part, for example, and used to provide context, such as start and end times, to other web parts on the page.

Event frame templates

Using event frame templates, you can define and standardize the related data (event frame attributes) associated with different types of events. You can use event frame attributes to provide additional context around the event that are useful for searches and reports. For example, downtime events often have a reason code that users want to search for or filter on during analysis of their downtime events.

You can also configure event frame attributes to reference process data in the context of the event. For example, a temperature excursion event is likely to have an attribute for the maximum temperature. You can configure event frames to record those values for you. Additionally, for each event type, you can configure an index for the most-searched attributes: this enables faster and easier searches on the PI AF server when you track several event types or millions of events.

Data references in event frame templates

When you create event frames dynamically with event frame templates, you typically need to reference elements, attributes, templates, or other related objects based on those event frame templates. Because each individual event frame occurs in a slightly different context, you can use substitution parameters to reference other PI AF objects dynamically, rather than static, absolute references. For example, you can use a downtime event frame template for any number of assets, such as a pump, motor, boiler, compressor, and so on.

In PI point data references, substitution parameters are useful for:

- Building tag name references based on hierarchy or other attribute values.
- Referencing other PI point attributes, such as from an event frame to the primary referenced element.

The main limitation of PI point data references is that the element attribute must also be a PI point data reference.

In String Builder data references, substitution parameters are useful for:

- Obtaining numeric values or string values when referenced element attributes do not have a PI point data reference.
- Creating a string by concatenating multiple values from referenced element attributes and other text.
- Passing a time context, including the end time of the event frame.
- Obtaining the name of referenced elements or their parents.

The limitation of String Builder data references is that they do not perform unit of measure conversions or perform aggregations or calculations.

A specific syntax is required to reference attributes from event frame templates, as described in Data references to attributes in the primary referenced element and Data references to attributes in other elements.

Video

For information on how to set up event frame templates, watch this video:

https://www.youtube.com/watch?v=0gKaw9p4FA0&rel=0

Primary referenced element

Each event frame references one or more elements. The event frame's **Referenced Elements** tab lists all elements that the event frame refers to. The main element referenced by the event frame is called the primary referenced element, and is indicated by a checkmark on the element icon:

By default, the first referenced element added is set as the primary referenced element, but you can change it by right-clicking on another element and selecting **Set as Primary Element Reference**.

> **Note:**
> If you delete the reference to the primary element, PI System Explorer does not automatically set a new primary referenced element. You must select a new primary referenced element manually.

Data references to attributes in the primary referenced element

In data references from an event frame template, you can refer to an attribute in an event frame's primary referenced element with the syntax:

```
.\Elements [.]|AttributeName
```

`.`	Indicates the current object, which is the event frame.
`\Elements`	Specifies the elements collection of the current object.
`[.]`	Represents the default object of the parent object, which is the primary referenced element.
`|AttributeName`	Specifies the name of the attribute.

For example, `.\Elements [.]|Flowrate` obtains the value of the `Flowrate` attribute from the primary referenced element.

Data references to attributes in other elements

In data references from an event frame template, you can reference other elements and attributes with the collection filters shown in the following table.

Collection filter	Example	Explanation
`@Category`	`.\Elements[@Category=MyCategory]`	Obtains the value of the first referenced element that is assigned to the `MyCategory` category.
`@Description`	`.\Elements[@Description=My Description]`	Obtains the value of the first referenced element that has the `My Description` description.
`@Index`	`.\Elements[@Index=3]`	Obtains the value of the third referenced element.
	`.\Elements[@Index=2]|FlowRate`	Obtains the value of the `FlowRate` attribute from the second referenced element.

Collection filter	Example	Explanation
@Name **Note:** You can use wild cards with **@Name**.	`.\Elements[@Name=Tank1]`	Obtains the value of the element named `Tank1`.
	`.\Elements[@Name=*2]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name ends in 2.
@ReferenceType	`.\Elements[@ReferenceType=Area]`	Obtains the value of the first referenced element that uses the `Area` reference type.
@Template	`.\Elements[@Template=MyTemplate]`	Obtains the value of the first referenced element that uses the `MyTemplate` template.
	`.\Elements[@Template=Pump Template]|%attribute%`	Obtains the value of the attribute that has the same name as this event frame attribute from the first referenced element that uses the `Pump Template`.
@Trait	`.\Elements[@Name=Tank*]|Attributes[@Trait=LoLo]`	Obtains the value of the attribute with a `LoLo` trait in the first referenced element whose name begins with `Tank`.
@Type	`.\Elements[@Name=Tank*]|Attributes[@Type=System.Int32]`	Obtains the value of the first attribute that has an `Int32` value type in the first referenced element whose name begins with `Tank`.
@UOM	`.\Elements[@Name=Tank*]|[@UOM=meter]`	Obtains the value of the first attribute that uses the `meter` unit of measure in the first referenced element whose name begins with `Tank`.

You can also combine collection filter criteria, as shown in the following examples:

Example	Explanation
`.\Elements[@Name=*2][@Category=Pump]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name ends in 2 and whose element category is `Pump`.
`.\Elements[@Name=Pump*][@Template=Pump Template]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name starts with `Pump` and whose element template is `Pump Template`.
`.\EventFrames[@Template=Machine Template]|Pressure`	Obtains the value of the `Pressure` attribute from the first child event frame whose template is `Machine Template`.

Example	Explanation

`.\EventFrames[@Template=Machine Template][@Index=-1]|Pressure`
or
`.\EventFrames[@Template=Machine Template][-1]|Pressure`
	Obtains the value of the `Pressure` attribute from the most recent child event frame whose template is `Machine Template`.
	Note:
	Beginning with PI AF 2017, you can use a negative number for an index from the end of a collection (for example -1 indicates the last item, -2 indicates the penultimate item).
	The **`@Index`** filter is optional if another filter is specified before the index filter.

Create event frame templates

Create and configure event frame templates to ensure consistent definitions for event frames in your asset structure.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, click **Event Frame Templates**.
3. To create a new event frame template, choose one of the following actions:
   - On the toolbar, click **New Template**.
   - Right-click **Event Frame Templates** and click **New Template**.
   - Right-click in the **Event Frame Templates** pane and click **New Template**.
4. In the **General** tab of the Event Frame Template Properties window, enter a unique name and a description for the event frame template in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the event frame template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing event frame template, which you select from the drop-down list. For more information on base templates, see Base and derived templates.
Severity	Select a severity setting for event frames based on the template. You can choose **None** (default), **Information**, **Warning**, **Minor**, **Major**, and **Critical**. **Note:** Event frames that have already been created with a specific severity setting are not updated automatically if the **Severity** setting in an event frame template is later changed.

Option	Description
Categories	Optional. Organize event frame templates by grouping them into element categories. To browse available categories or to create a new category, click .
Default Attribute	Optional. After you have created attributes for the template, you can select a default attribute from the drop-down list. For more information, see Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current local start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCSTARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute template, represented by the path. ◦ **%ELEMENT%** The name of the primary referenced element of the object's event frame. ◦ **%ELEMENTDESCRIPTION%** The description of the primary referenced element of the object's event frame. You can also use other naming pattern substitution parameters. For example, if you want an event frame generation analysis name included in the naming pattern, enter `%ANALYSIS%` as a substitution parameter. For a complete list of naming pattern substitution parameters, see Naming patterns. Each element derived from the template will have a unique, identifiable name. To ensure that new elements created from the template have an incremental number, enter * at the end of any pattern you enter here. Naming patterns are re-evaluated on an event frame when the **Start time**, **End time**, or primary referenced element is modified after it has been created. It will not be re-evaluated when loaded from another application. To force an element or event frame to re-evaluate its naming pattern, right-click the object and click **Reevaluate Naming Pattern**. Some substitution parameters may not be applied when a derived event frame is created. To ensure a derived event frame's name fully reflects the naming pattern, right-click the event frame and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters. If left blank, PI System Explorer uses the following rules: ◦ Event frames that are generated by event frame generation analyses use a default naming pattern of: `%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff%` ◦ Event frames that are created by other methods use the name of the event frame template and the current date: `%TEMPLATE%yyyyMMdd`

Option	Description
Allow Extensions	Select this check box to enable additional attributes to be defined for an event frame, beyond those defined in its template. For more information, see Extensions. **Note:** Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.
Can Be Acknowledged	Select this check box to enable users to acknowledge an event frame. For more information, see Acknowledgement of event frames. **Note:** This setting only affects event frames that are created after a template is created or modified.
Location	Click this link if you wish to specify location attribute traits for the event frame template. For more information, see Set location attribute traits.
Reason	Click this link if you wish to set up reason attribute traits for the event frame template so that users can select reason codes for downtime, excursions, and other events. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the event frame template beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.

Change event frame templates

You can change the template on which an existing event frame is based, or add a template to an event frame that is not currently based on a template.

Before you start

Exercise caution when you change an event frame template, since there can be unintended consequences. For example, attributes could be deleted if they were defined by the old template and are not present in the new template. Be sure that attributes with the same name in both the old and new template have the same data type.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, right-click the **Event Frame Searches** collection and click **New Search**.
3. In the Event Frame Search window, enter criteria to locate the event frame that you want to change.
4. Right-click the appropriate event frame in the browser and click **Convert** > **Change Template**.

5. In the Choose Event Frame Template window, select the desired template from the **Event Frame Template** list.
   a. Optional. To display only templates in a specific category, select a category from the **Templates of category** list.
6. Click **OK**.

Create event frames

Although you typically use event-frame-generation analyses or PI Event Frames Generator (PI EFGen) to create event frames, you can create an event frame manually. It is simply not recommended.

Procedure

1. In the navigator, select **Event Frames**.
2. To create a new event frame, choose one of the following actions:
   - On the toolbar, click **New Event Frame**.
   - In the Event Frames browser, right-click the **Event Frame Searches** collection and click **New Event Frame**.
   - In the viewer, right-click and click **New** > **New Event Frame**.
3. In the **Event Frame Template** section of the Choose Event Frame Template window, select an event frame template and click **OK**.

   The new event frame is displayed in the viewer with four tabs for configuring the event frame.
4. In the **General** tab, enter a unique name and a description for the event frame in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the event frame. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Template	Displays the template that you selected when you created the event frame. To review the template properties, click .
Severity	Select a severity setting for the event frame. You can choose **None** (default), **Information**, **Warning**, **Minor**, **Major**, and **Critical**.
Start time	Defaults to current date and time. Press F2 or click to open the Date and Time Picker window where you can select a different start time.
End time	Defaults to no date or time. Press F2 or click to open the Date and Time Picker window where you can select a different end time.
Categories	Optional. You can organize event frames by grouping them into element categories. To browse available categories, click .

Default Attribute	Optional. This field is read-only if the event frame is based on a template. After you have created attributes for the event frame in the **Attributes** tab, you can select a default attribute from the drop-down list. For more information, see Default attribute.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Annotations	Click this link if you wish to enter a comment and/or add a file attachment specific to the event frame. For more information, see Add annotations to event frames.
Location	Click this link if you wish to specify location attribute traits for the event frame. For more information, see Set location attribute traits.
Reason	Click this link if you wish to set up a reason attribute trait for the event frame. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the event frame beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Optional. You can choose from the following actions, as needed.

Option	Description
Capture values	To improve performance, you can save event-frame attribute values in PI AF rather than have data references calculated and executed. For more information, see Value capture for event frames and transfers.
Lock	After the underlying action for an event frame has completed, such as batch completion, you can click this link to prevent further changes to it. For more information, see Lock event frames or transfers. **Note:** Only Administrator users can unlock an event frame.
Acknowledge	If the event frame is based on a template where both the **Can Be Acknowledged** setting and the annotation access right are enabled, click this link to indicate that you have acknowledged the event frame. For more information, see Acknowledgement of event frames.

7. Edit the remaining tabs as needed.
   - To add attributes, see Create attributes on event frames.
   - To set a primary referenced element, see Primary referenced element.

Create attributes on event frames

Create attributes on event frames in the same way as on elements.

Procedure

1. In the Event Frames browser, find and select the parent event frame.
2. In the viewer, select the **Attributes** tab.
3. If no attributes exist, click the **New Attribute** link. Alternatively, click **New Attribute** on the toolbar.

   Note:
   If the event frame is based on a template, you cannot add an attribute unless the template allows extensions.

4. In the **Name** field, enter a unique name for the attribute.
5. In the **Description** field, enter a description for the attribute.
6. In the **Properties** field, select attribute properties as needed. For more information, see Attribute properties.
7. Optional. In the **Categories** field, assign the attribute to a category. To browse available categories, click .
8. In the **Default UOM** field, select the unit of measure for the attribute.
9. In the **Value Type** field, select the data type of the attribute.
10. To assign a value to the attribute, choose from the following actions.

To ...	Do this ...
Enter a static value	Type a value in the **Value** field.
Derive or calculate a value from a data reference	a. Select a data reference type from the **Data Reference** list. b. Click **Settings** to configure the data reference. For details about configuring a data reference, see Configuration of data references.

11. Optional. When the **Use DisplayDigits for Attribute and Attribute Template values** system option is selected, you can override the default DisplayDigits property value of -5 and specify the number of digits to display in the **Display Digits** field. Only numeric values in the **Value** field are affected. For attributes that are not based on a template, the precision of PI point data references is obtained from the PI point itself.
    - Enter a negative number up to -20 to indicate the total number of digits to display.
    - Enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

    For more information, see Control the display of attribute and attribute template values and PI System Explorer customization options.
12. To save your work, press CTRL-S or click **Check In**.

Value capture for event frames and transfers

To display values for an event frame or transfer, PI AF executes data references to retrieve associated attribute values for the time period. You can, however, use **Capture Values** to save those values in a PI AF database table (`AFEventFrameAttributeValue`). Doing this can improve performance, since it is faster to display the saved values than to retrieve them.

You can also use **Capture Values** to ensure attribute values displayed for event frames are the same at any future point as when they were captured. Additionally, **Capture Values** provides a way to preserve values that might not be available later, such as streaming data that is not persisted long-term.

Value recapture

Automatic recapture occurs whenever the start time and end time of a captured event frame or transfer is modified.

Whenever you add new attributes to an event frame or transfer, you must recapture values to ensure that values are also captured for the new attributes.

Note that when you recapture values, data loss can occur if a data reference has changed since the initial value capture: For example, if values in a table accessed by a table lookup data reference have been modified.

Capture values

You capture values to save event frame or transfer attribute values to a table in the PI AF database. This can improve performance since PI AF does not execute any data references.

> **Note:**
> If you add new attributes to event frames or transfers with captured values, you should recapture those values to ensure that values are also captured for the new attributes.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, choose from the following actions.

To capture values for ...	Do this ...
All transfers or parent event frames in a collection	a. Right-click a search or recent collection. b. Click **Capture or Recapture Values**. Alternatively, follow these steps. a. Click a search or recent collection. b. In the viewer, right-click below the list of event frames or transfers. c. Click **Capture or Recapture Values**.

Child event frames in a collection	a. Click a search or recent collection. b. In the viewer, click ⊞ beside each event frame for which you want to capture values. The child event frames are displayed. c. In the viewer, use Windows selection keystrokes to select event frames and child event frames: ▪ To select contiguous objects, press SHIFT +<click>. ▪ To select non-contiguous objects, press CTRL+<click>. ▪ To select all displayed objects, press CTRL +A. d. Right-click a selected event frame. e. Click **Capture or Recapture Values**.
A single event frame	a. Expand a search or a recent event frame collection. b. Select the event frame. c. Choose from the following actions. ▪ In the **General** tab, click the **Capture Values** link. ▪ Right-click and click **Capture or Recapture Values**. **Note:** For subsequent value captures, the link changes to **Recapture Values**.
A single transfer	a. Expand a search or a recent transfer collection. b. Select the transfer. c. Right-click and click **Capture or Recapture Values**.

3. In the Capture or Recapture Values window, check the status and click **Close**.

Results

A Has Captured Values icon (📌) is displayed in the viewer beside each event frame or transfer that has a captured value.

Lock event frames or transfers

After the underlying action for an event frame or transfer has completed (for example, after a batch has finished), the event frame or transfer is still open and can continue to receive data. You can lock an event frame or transfer, preventing further changes to it.

Unlocking an event frame or transfer requires Administrator permission. Only users with Administrator permissions on an object have the ability to unlock it.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, open the **Event Frame Searches** (or **Transfer Searches**) collection that contains the object you want to lock.
3. Right-click the object and click **Lock**.
4. To verify that the object is locked, select the event frame or transfer search collection that contains the object.

   The object is displayed in the viewer with next to it in the Lock column.

Acknowledgement of event frames

In PI AF 2016, you can require that an event frame be acknowledged. The acknowledgement feature is used by notifications and PI AF client applications, such as PI Vision. The event-frame acknowledgement feature replaces the acknowledge notification functionality in the legacy version of Notifications.

- For more information on setting up event-frame acknowledgement in notifications, see Configuration of notification rules for analyses or event frames.
- For more information on viewing and acknowledging event frames using PI Vision, see the PI Vision topic "View event details and annotate events" in Live Library (https://livelibrary.osisoft.com).

Security for event-frame acknowledgement

A user needs to have a PI AF identity for which the Annotate permission (an) is enabled. For more information on access rights, see PI AF access rights.

Setup of event-frame acknowledgement

You set up an acknowledgement by selecting the **Can Be Acknowledged** check box as you configure an event frame template. Any event frame that is based on that template displays an **Acknowledge** link on the **General** tab. In event frames based on event frame templates where the **Can Be Acknowledged** check box is not selected, the **Acknowledge** link is inactive. You also cannot acknowledge event frames that are not based on an event frame template.

> **Note:**
> The **Can Be Acknowledged** setting only affects event frames that are created after a template is created or modified.

Search for event-frame acknowledgements

You can search for event frames that can be acknowledged, as well as those that have been acknowledged. You review acknowledgement status in the Viewer for the Event Frame Searches collection, where an Acknowledged column is displayed ( ). Possible values are:

	Is not acknowledged
	Is acknowledged
blank	Not acknowledgeable

To group event frames by their acknowledged status, click the column icon.

Acknowledge event frames

You acknowledge event frames in the Event Frames browser or in the **General** tab of a selected event frame in the Event Frames viewer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click an Event Frame Search collection. The contents of the collection are displayed in the viewer.
3. Optional. To group event frames by their acknowledged status, click the ⚠ column icon.
4. Choose from the following actions.

To ...	Do this ...
Acknowledge an event frame	a. Choose one of the following actions: ▪ Right-click an event frame in the viewer and click **Acknowledge** on the context menu. ▪ Double-click an event frame in the viewer and, on the **General** tab, click the **Acknowledge** link. b. Click **Yes** in response to the **Are you sure you want to acknowledge *Event_Frame_Name*?** prompt. c. The following changes occur: ▪ In the Event Frame Search collection that is displayed in the viewer, the Acknowledged (⚠) column changes to OK status. ▪ On the **General** tab, the **Acknowledged** field is displayed with a time and date stamp. The user ID of the user who made the acknowledgement is displayed in the **By** field.
Review an acknowledgement	a. In the Event Frame Search collection that is displayed in the viewer, double-click an event frame that has an OK status. b. On the **General** tab, review the time and date stamp in the **Acknowledged** field, as well as the user ID of the user who made the acknowledgement.

Transfers

Transfers are a type of event frame. They mark movement of material in discrete quantities. They have a start time and an end time. Transfers are unique in a model because they are temporal and appear in a model only when a transfer of material has taken place. For example,

use transfers to track material movements in and out of the facility, track raw materials used in the process and finished product being stored, and track tank-to-tank material transfers.

Topics in this section

- Create transfer templates
- Create transfers
- Create transfer ports

Create transfer templates

Create and configure transfer templates to ensure consistent definitions for transfers in your asset structure.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, click **Transfer Templates**.
3. To create a new transfer template, choose one of the following actions:
   - On the toolbar, click **New Template**.
   - In the **Transfer Templates** pane, click **New Template**.
   - Right-click **Transfer Templates** and click **New Template**.
4. In the **General** tab of the Transfer Template Properties window, enter a unique name and a description for the transfer template in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the transfer template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing transfer template, which you select from the list. For more information on base templates, see Base and derived templates.
Categories	Optional. Organize transfer templates by grouping them into element categories. To browse available categories or to create a new category, click .
Default Attribute	Optional. After you have created attributes for the template, you can select a default attribute from the list. For more information, see Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%TIME:yyyy-MM-dd HH:mm:ss.fff%** The current local time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current local start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%UTCSTARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute template, represented by the path. ◦ **%SOURCE%** Name of the source element for the transfer. ◦ **%DESTINATION%** Name of the destination element for the transfer. You can also use other naming pattern substitution parameters. For a complete list of naming pattern substitution parameters, see Naming patterns. Each transfer derived from the template will have a unique, identifiable name. To ensure that new transfers created from the template have an incremental number, enter * at the end of any pattern you enter here. Naming patterns are re-evaluated on a transfer when the **Start time** is modified after it has been created. It will not be re-evaluated when loaded from another application. To force a transfer to re-evaluate its naming pattern, right-click the object and click **Reevaluate Naming Pattern**. Some substitution parameters may not be applied when a derived transfer is created. To ensure a derived transfer's name fully reflects the naming pattern, right-click the transfer and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters. If left blank, PI System Explorer uses the name of the transfer template and the current date: `%TEMPLATE%yyyyMMdd`

Option	Description
Allow Extensions	Select this check box to enable additional attributes to be defined for a transfer, beyond those defined in its template. For more information, see Extensions. **Note:** Categories are not affected by whether the **Allow Extensions** check box is enabled.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Reason	Click this link if you wish to set up a reason attribute trait for the transfer template. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the transfer template beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.
7. Optional. Use the **Ports** tab to define additional ports that define end-points for connections between transfer elements, or to modify the default **In** and **Out** ports. See Create transfer ports.

Create transfers

You create a transfer in the Event Frames browser.

Before you start

Default **In** and **Out** transfer ports with Boundary and Node connection types are automatically created for every transfer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click the **Transfer Searches** (or **Recent Transfers**) collection.
3. To create a new transfer, choose one of the following actions:
   - On the toolbar, click the **New Transfer** button.
   - In the **Event Frames** browser, right-click the **Transfer Searches** collection and click **New Transfer**.
   - In the viewer, right-click and click **New Transfer**.
4. In the Choose Transfer Template window, choose an existing template for the transfer. Although not recommended, you can also choose **<None>**.
   a. Optional. To assign the transfer to a category, select one from the **Templates of category** list.
5. Click **OK**.

   The new transfer is displayed in the viewer with three tabs for configuring the transfer. In the **General** tab, a default name for the new transfer is displayed based on the naming pattern, as well as a default start time.

6. Optional. Replace the default name with a unique name and enter a description for the transfer in the **Name** and **Description** fields.
7. Optional. You can organize objects by grouping them into categories. To browse the available categories, click . See Categorization of objects for more information.
8. In the **Start time** and **End time** fields, enter the start and end times for when the transfer takes place. To browse for a date, click .
9. In the **Source** and **Port** fields, select the element and port providing the material of the transfer.
10. In the **Destination** and **Port** fields, select the element and port receiving the transfer.
11. If the transfer is based on a template, a read-only value is displayed in the **Default Attribute** field. However, if you are creating a transfer manually, you can select a default attribute after you have added attributes in the **Attributes** tab.
12. Optional. To add an annotation to the transfer, click **Annotations**. For more information, see Add annotations to transfers.

After you finish

If the **Allow Extensions** check box is selected in the transfer template or you are creating a transfer manually, follow these additional steps.

- Use the **Attributes** tab to define an attribute for each property or data item. See Create attribute templates.
- Use the **Ports** tab to define additional ports that define end-points for connections between transfer elements, or to modify the default **In** and **Out** ports. For more information, see Create transfer ports.

Create transfer ports

Use the **Ports** tab to review settings for default **In** and **Out** transfer ports.

Before you start

You can only modify or add ports for a transfer derived from a template if **Allow Extensions** is enabled on the transfer template.

Procedure

1. To create a new transfer port or review existing ports, choose one of the following actions:
   - In the Event Frames browser, expand a **Transfer Searches** collection. Select a transfer, or right-click a transfer and click **Properties**.
   - In the Library browser, expand **Transfer Templates**. Select a transfer template, or right-click a transfer template and click **Properties**.
2. Click the **Ports** tab.
3. Choose from the following actions.

To ...	Do this ...
Create a new port	a. Choose one of the following actions. ▪ On the toolbar, click **New Port**. ▪ Right-click in the viewer and click **New Port**. b. Choose one of the following actions. Refer to Transfer Ports tab properties for descriptions of required values. ▪ Edit the values in the **Ports** columns directly. ▪ Right-click the port and click **Properties**. Enter values and click **OK** to save your changes.
Modify an existing port	Choose one of the following actions. Refer to Transfer Ports tab properties for descriptions of required values. ◦ Edit the values in the **Ports** columns directly. ◦ Right-click the port and click **Properties**. Edit values and click **OK** to save your changes.
Specify a default port	a. Right-click a port that displays the icon. b. Select **Set as Default Port**.
Modify a default port	a. Right-click a port that displays the icon. b. Select **Properties**. c. In the Port Properties window, clear the **Default Port** check box. d. Click **OK**.

4. Check in your changes.

Transfer Ports tab properties

The **Ports** tab for a transfer contains the following columns.

Field	Description
Name	The name of the port.
Description	Optional. The description of the port.
Port Type	Indicates the type of port: **Input**, **Output**, or **Undirected** (for meters, for example).
Allowed Categories	The categories to which the port belongs. To assign categories, click and select categories in the Categorize window.
Maximum Connections	The maximum number of connections that can be made to the port. The default number of connections for a transfer is 1. A value of zero indicates an unlimited number of connections.
Connection Type	The type of connections to which the port can connect. The default connection types for a transfer are **Boundary** and **Node**. For more information on connection type values, see Element types in models.
Allowed Template	Defaults to all element templates, but you can select a specific template from the list.

Annotations in PI AF

Beginning with PI AF 2016, the annotation feature is used by client applications such as PI Vision, but can easily be used by administrators to make notes on the following objects:

- Case
- Element
- Event frame
- Model
- Transfer

File attachments

Users can attach a single file to an annotation. By default, the following file types are allowed:

File type	Allowed extension
MS Office	csv, docx, pdf, xlsx
Text	rtf, txt
Image	gif, jpeg, jpg, png, svg, tiff
ProcessBook	pdi

The maximum file size defaults to 10MB.

Administrators can use the PI AF Diagnostics utility (`afdiag` located in the `\PIPC\AF` folder) to specify the file types that can be attached to annotations (with the **`FileExtensions`** parameter), as well as the maximum file size (with the **`FileMaxLength`** parameter).

Security for annotations

A user needs to have a PI AF identity for which the Annotate permission (**`an`**) is enabled. For more information on access rights, see PI AF access rights.

Topics in this section

- Element annotations
- Event frame annotations
- Transfer annotations

Element annotations

Users can annotate an element, as well as upload a file to link to an annotation.

Setup of element annotations

You set up an annotation by opening an Annotations window for a specific element that you have selected in the Elements browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Review of element annotations

You review annotation status for an element in the Elements viewer, where an Is Annotated column is displayed ( ).

To view the content of an annotation, move the mouse pointer over the icon beside an element to see the annotation text. If a file is attached, the name of the file is also displayed.

Add annotations to elements

You can add one or more annotations to an element.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, choose one of the following actions.
   - Click the **Elements** collection.
   - Navigate to the specific element you want to annotate.
3. To add a new annotation, choose one of the following actions.
   - In the Elements browser, right-click an element and click **Annotate**.
   - If an element is selected in the Elements browser, click the **Annotations** link on the **General** tab in the viewer.
   - If the **Elements** collection is selected in the browser, right-click an element in the viewer and click **Annotate**.

     Note:
     To add another annotation to a previously annotated element in the viewer, double-click beside the element.
4. In the Annotations window, click **New Annotation**.
5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.
6. To attach a file to the annotation, click **Add Attachment**.
   a. In the Open window, navigate to the directory where the file you wish to upload is located.
   b. Select the file and click **Open**.
   The filename is displayed in the **Attachment** field.
7. Click **Close**.

Results

When elements are listed in the Elements viewer, an Is Annotated column is displayed beside the element. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Event frame annotations

Users can annotate an event frame, as well as upload a file to link to an annotation.

Setup of event frame annotations

You set up an annotation by opening an Annotations window for a specific event frame that you have selected in the Event Frames browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Search for event frame annotations

You can search for event frames that have been annotated. You review annotation status in the viewer for an Event Frame Searches collection, where an Is Annotated column is displayed ( ).

To group event frames by their annotation status, you click the column icon.

Add annotations to event frames

You can add one or more annotations to an event frame.

Before you start

To annotate a locked event frame, a user with Administrator permission must first unlock it.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click an **Event Frame Searches** collection. The contents of the collection are displayed in the viewer. Event frames with existing annotations display an Is Annotated icon ( ) beside them.
3. To add a new annotation, choose one of the following actions.
   - In the viewer, right-click an event frame and click **Annotate**.
   - In the viewer, double-click an event frame and click the **Annotations** link on the **General** tab.
   - In the Event Frames browser, right-click an event frame in an event frame search collection and click **Annotate**.
   - To add another annotation to a previously annotated event frame, double-click beside the event frame.
4. In the Annotations window, click **New Annotation**.
5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.

6. To attach a file to the annotation, click **Add Attachment**.

a. In the Open window, navigate to the directory where the file you wish to upload is located.

b. Select the file and click **Open**.
The filename is displayed in the **Attachment** field.

7. Click **Close**.

Results

In the Is Annotated column, is displayed beside the event frame. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Transfer annotations

Users can annotate a transfer, as well as upload a file to link to an annotation.

Setup of transfer annotations

You set up an annotation by opening an Annotations window for a specific transfer that you have selected in the Event Frames browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Review transfer annotations

You can review annotation status in the viewer for a Transfer Searches collection, where an Is Annotated column is displayed ( ).

To group transfers by their annotation status, you click the column icon.

Add annotations to transfers

You can add one or more annotations to a transfer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click a **Transfer Searches** collection. The contents of the collection are displayed in the viewer. Transfers with existing annotations display an Is Annotated icon ( ) beside them.
3. To add a new annotation, choose one of the following actions.
   - In the viewer, right-click a transfer in the search result list and click **Annotate**.
   - In the viewer, double-click a transfer and click the **Annotations** link on the **General** tab.
   - In the Event Frames browser, right-click a transfer in a **Transfer Searches** collection and click **Annotate**.
   - To add another annotation to a previously annotated transfer, double-click beside the transfer.
4. In the Annotations window, click **New Annotation**.

5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.

6. To attach a file to the annotation, click **Add Attachment**.

   a. In the Open window, navigate to the directory where the file you wish to upload is located.

   b. Select the file and click **Open**.
   The filename is displayed in the **Attachment** field.

7. Click **Close**.

Results

In the Is Annotated column, is displayed beside the transfer. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Asset analytics

Asset analytics is the feature in PI Asset Framework (PI AF) that you use to create and manage analyses. An analysis reads values of PI AF attributes, performs calculations, and writes results to other attributes. Within PI System Explorer, you can access asset analytics when you work with elements and element templates.

Topics in this section

- Introduction to analyses
- Expression analyses
- Rollup analyses
- Event frame generation analyses
- SQC analyses
- Sample analyses
- Backfilling and recalculation of analyses
- Management of analyses in a PI AF database
- Expression functions reference
- Steam functions for analysis expressions
- Asset analytics native integration of external systems
- PI Analysis Service management

Introduction to analyses

An analysis performs calculations on a specified schedule. An analysis takes existing PI AF attribute values as inputs and produces new outputs, either new calculated values or new event frames. Every analysis has an associated element or element template.

The following types of analyses are available:

- **Expression**

  Calculates one or more output values from specified functions, operators, and input values.

- **Rollup**

  Calculates standard statistical functions for a group of selected attributes. This group is selected from attributes on an element or from the set of all attributes on its sub-elements.

- **Event frame generation**

  Starts or ends event frames based on specified conditions.

- **SQC**

  Uses Statistical Quality Control (SQC) methods to monitor that attribute values lie within predetermined boundaries.

All analyses can be created from analysis templates with either `Enabled` or `Disabled` status; by default, analyses will be enabled when created. To create an analysis with `Disabled` status, make sure to clear the **Enable analyses when created from template** check box in the analysis template.

For event frame generation analyses and SQC analyses with event frame outputs, you can create a notification rule template by clicking **Create a new notification rule template for *Analysis Template Name***.

For expression analyses and event frame generation analyses, you specify inputs in expressions.

For event frame generation analyses (running PI Analysis Service 2017 R2 or later versions), expression and rollup analyses, you can map analysis outputs to PI AF attributes. If you map outputs to attributes configured as PI point data references, the analysis saves the history of the output to a PI point. OSIsoft recommends that you save analysis outputs to PI points. You can use the PI point in visualization tools to view the trend of the analysis. Saving outputs to PI points also results in better PI AF performance than saving outputs to attributes without data references.

Note:
PI Analysis Service does not use exception reporting while writing to output configured as PI point data references. Although exception settings for the output PI point are not used, compression settings for the output PI point will be used to determine the outputs that are written to PI Data Archive.

Topics in this section

- Configure and manage analyses on an element
- Analysis templates
- Updates of analyses and analysis templates
- Excluded attributes in analyses
- Analysis scheduling
- Expressions for analyses
- Future data and analyses

Configure and manage analyses on an element

A newly-created analysis is listed in the table in **Analyses** tab. You can see its current status, whether or not it is associated with a template, its analysis type, name, and backfilling/recalculation status. Hover over items or icons in the list to see descriptive tooltips. To manage multiple analyses in the **Management** pane, see Management of analyses.

To view the expression and scheduling for an analysis, select it from the **Analyses** viewer.

You can perform these operations from the **Analyses** viewer:

- **Create a new analysis**

  You can click (New Analysis) at the top of the viewer, or right-click anywhere in the viewer and click **New**.

- **Enable or disable a selected analysis**

  You can select an analysis and click (Enable Analysis) or (Disable Analysis) at the top of the viewer.

- **Choose columns to display**

  You can right-click any column heading and select headings to display.

- **Preview results**

  You can preview results for an analysis to see how it operates over a specified time range by right-clicking an analysis and selecting **Preview Results**. You can export the results table to a spreadsheet or you can copy selected rows from the results table into other applications.

  For information on viewing trend charts for output times that are specified relative to trigger times, see Trend charts in Preview Results.

- **Backfill or recalculate data for an analysis**

  You can execute an analysis over an earlier time period to backfill or recalculate data, as described in Backfill or recalculate data for an analysis.

  An expression, rollup, or SQC analysis can backfill or recalculate data if the analyses outputs are mapped to PI point attributes. You can backfill the missing data in a time range and retain existing data, or you can recalculate, which deletes and recalculates data in the time range.

  An event frame generation analysis can also recalculate event frames over a specified time period. The recalculation process, however, automatically deletes all existing event frames for that time period, as well as annotations on affected event frames.

  In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. Note that recalculation is only available for PI Analysis Service 2016 R2 or later. In addition, PI Data Archive that stores PI points must be running version 2016 R2 or later.

- **Enable automatic recalculation for an analysis**

  From PI AF 2017 R2, you can automatically recalculate analyses. Although automatic recalculation is globally enabled by default, you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation.

  An analysis listed with means automatic recalculation is enabled. See Configure automatic recalculation of analyses and analyses templates for more information.

- **Go to template**

  An analysis listed with alongside is based on an analysis template. You can open the template directly from the analysis by right-clicking the analysis and selecting **Go to Template**.

  > **Note:**
  > You can go to the analysis directly from the attribute by right-clicking the attribute and clicking **Go to Analysis**. If there is no analysis configured on the attribute, you will see **New Analysis** instead.

- **Audit Trail Events**

  You can open the audit log directly from the analysis by right-clicking the analysis and selecting **Audit Trail Events**. For more information on audit trail, see Track PI AF changes with Audit Trail.

- **Reset to template**

  Only scheduling changes can be made to an analysis that is based on an analysis template. To revert an analysis' scheduling to template scheduling, right-click the analysis and click **Reset to Template**.

- **Convert to template**

  If the element for an analysis is derived from an element template, you can convert the analysis to an analysis template on that element template by right-clicking the analysis and selecting **Convert to Template**. This feature enables you to develop and debug an analysis against a specific element before generalizing it as a template.

  > **Note:**
  > If any of the analysis outputs write to a specific PI point, you will be prompted to choose how to specify the PI point in the analysis template to ensure the outputs of derived analyses write to unique PI points.

To create specific types of analyses, see:

- Create an expression analysis
- Create a rollup analysis
- Create an event frame generation analysis
- Create an SQC analysis

Analysis templates

To apply an analysis to an element, you specify the element directly as you create the analysis. However, to apply an analysis to a group of elements (created from the same element template), it is much more efficient to use an *analysis template* than to apply the analysis individually to each element.

An analysis template defines the form of an analysis. Analyses derived from it are all similar but have specific input and output attributes. Analysis templates let you take advantage of your PI AF hierarchy. Every element derived from an element template automatically acquires analyses from its analysis templates.

You create an analysis template on an element template in much the same way you create an analysis on an element. When you add or change an analysis template on an element template, those changes propagate to every element derived from the element template. Deleting an analysis template deletes all analyses derived from it, except for analyses tied to notifications.

With PI Server 2015 R2 and later versions, while creating an analysis template on an element template that is derived from a base element template, you can choose input attributes from attribute templates that are defined on derived as well as base element templates. Similarly, you can map analysis outputs to derived as well as base template attributes.

> **Note:**
> An analysis derived from an analysis template cannot be removed from an element directly.

To create specific types of analysis templates, see:

- Create an expression analysis template
- Create a rollup analysis template
- Create an event frame generation analysis template
- Create an SQC analysis template

Updates of analyses and analysis templates

As soon as a new analysis is checked in, it is automatically enabled and available to run. Any change that is checked in is immediately picked up by all the analyses affected by it, even if they are running. Changes that can affect an analysis include:

- Edits made directly to the analysis
- Changes to an element that impact its analyses, including adding or deleting an analysis
- Changes to an element template, which propagate to all elements that come from the template, including adding or deleting an analysis template
- Changes to an analysis template, which propagate to all analyses that come from the template
- Changes to an event frame template used by an analysis; new event frames are based on the updated template

Excluded attributes in analyses

In PI Asset Framework 2015, excluded attributes were introduced. For more information on excluded attributes, see Excluded attribute property. Beginning with PI AF 2018, behavior and handling of excluded attributes have changed in expression, event frame generation and SQC analyses.

Prior to 2018, analyses configured with excluded attributes would return **Calc Failed** when evaluated and go into error. Beginning with PI AF 2018, such analyses will be suspended and marked by an icon (⏸) in the Analyses Viewer. If you hover over the icon, you will see tooltip with more information.

If you do not want the analyses with excluded attributes to go into a suspended state, you can validate the excluded attribute with an expression function **BadVal**. Assuming 'att1' is an excluded attribute, below is an example of how **BadVal** could be used to prevent the analysis from being suspended. In this case, the analysis will continue to run:

```
If BadVal('att1') Then ('att2') Else ('att1' + 'att2)
```

> **Note:**
> Any change in the configuration of the excluded attribute in the analysis is automatically picked up by PI Analysis Service. For example, if you decide to change the property of 'att1' to be no longer excluded and check it in, the service will detect it and run the analysis with 'att1.'

When you select **Event-Triggered** as a scheduling option, you are required to select a triggering input. Note that if the excluded attribute is the only available triggering input, the analysis will be suspended whether or not you validate it with **`BadVal`**. It is because analyses do not trigger on excluded attributes. Make sure that you have other inputs to trigger on in your analyses. For more information, see Analysis scheduling.

Analysis scheduling

Each analysis requires you to specify scheduling. Scheduling indicates when to evaluate an analysis automatically. There are two types of scheduling:

- **Periodic**

  With periodic scheduling, evaluation occurs based on clock time. You can specify a specific time for evaluation each day or the time between evaluations. With PI Server 2015 R2 and later versions, the maximum frequency you can set for analyses is 120 times per second.

  Note:

  - If you have analyses that are previously configured to run more than 120 times per second, you must reduce the period to less than the allowed maximum, else they will generate errors while running with PI Server 2015 R2. Similarly, if you are setting periodic scheduling outside of the PI System Explorer user interface, make sure that the period is set to less than the allowed maximum.
  - If the period is not divisible by the 24, 7 minutes for example, the evaluation time will reset at the beginning of each day. Let's say you have scheduled the analysis to run every 7 minutes and the last evaluation was at 11:56pm. The next evaluation will be at midnight (4 minutes later), not at 12:03am the next day.

- **Event-Triggered**

  With event-triggered scheduling, evaluation occurs based on events. You can specify one or more input attributes that trigger an evaluation whenever the attribute value changes.

  Note:
  When auto-backfilling is enabled, triggering events with time stamps before analysis service start-time are ignored for the purpose of real-time evaluation. For more discussion on real-time evaluation and auto-backfilling, including the setting of the **`AutoBackfillingEnabled`** configuration parameter, see Analysis service configuration.

By default, analyses use event-triggered scheduling, triggered on changes to any input.

Many factors affect the speed that an analysis runs and writes output values:

- System architecture, including inputs and outputs
- Network configuration
- Load and performance of different computers

Monitor and test your analyses, especially those scheduled at high frequencies, to ensure the system resources support the configuration.

Output time stamps for analyses

For any scheduled analysis, the default time stamp for output values is the *trigger time.* For periodic scheduling, the trigger time is the scheduled evaluation time, and for event-triggered scheduling, the trigger time is the time that a specified attribute changes values.

You can specify the time stamp of analysis output values. By specifying an output time stamp, you can:

- Produce results that are time-stamped at a specified offset from the trigger time.
- Produce results that are time-stamped at a particular time each day, regardless of when the analysis is actually evaluated. For example, you can ensure a 6:00 p.m. time stamp for the output of an analysis that calculates results for a daily production shift ending at 6:00 p.m., even if the analysis is not actually evaluated until 10:00 p.m.
- Repeatedly evaluate an analysis to calculate a forecast value for a particular date and time, as input conditions change. The results of each evaluation have same time stamp: the analysis output values reflect results from the most recent evaluation.

Specify time stamp for analysis outputs

Use the Advanced options window in an analysis to specify the time stamp of analysis output values.

Procedure

1. In the Elements browser, select an element, and in the viewer, click the **Analyses** tab.
2. From the list of analyses, select an analysis, and then click **Advanced** in the scheduling area near the bottom of the window.
3. In the Advanced options window, specify the output time stamp:
   - **Trigger Time**

     Default value. The clock time that a schedule specifies or that an input value changes.
   - **Execution Time**

     The clock time at which the analysis calculates the value. This differs from the trigger time when the system has a lag time or when there is a configured calculation wait time (see **`CalculationWaitTimeInSeconds`** in Analysis service configuration).
   - **Relative to Trigger Time**

     A time specified by a PI time expression. Enter a valid time expression, such as a time relative to the trigger time or a fixed time. In the expression, the current time and implied reference time is the trigger time (that is, the clock time that the schedule specifies or that an input value changes). If you enter a fixed time, all output values from the analysis will have the entered time stamp. When not specified, default unit of measure for relative time is hours.

     For information on viewing trend charts for output times that are specified relative to trigger times, see Trend charts in Preview Results.
   - **Variable**

Use the value of a variable as an output timestamp. **Variable** must be a timestamp of value type `datetime`. If this value is calculated from an expression, PI Analysis Service uses the calculated value for each trigger events as the output timestamp. This option is only available for expression analyses.

You can preview your results by right-clicking the analysis and selecting **Preview Results**. For more information, see Trend charts in Preview Results.

Trend charts in Preview Results

When you right click an expression or rollup analysis and select Preview Results, you see the trend chart with input and output attribute values at different trigger times.

If you select **`Relative to Trigger Time`** and specify '*-8h' as output time stamp for example, you see two trend charts displayed, one for input and the other for output. The input chart shows the same time range as the configured start and end time; however, the output chart shows the time that is calculated using the output time stamp override to the trigger time. Each trend chart shows a maximum of four input and four output attributes; currently, the input or output attributes that are displayed on the chart are automatically selected by PI Analysis Service.

In the table, the column `Output Time` indicates the output time calculated with the time stamp override.

Note:
The Preview Results charts may depict data that differs from the actual results of the PI Analysis Service. For example, if the same timestamp data is overwritten several times, the Preview Results table and charts show each written value of the data point, but the PI Analysis Service only uses the last written value.

Expressions for analyses

For expression analyses and event frame generation analyses, you specify inputs and calculations in expressions. In PI System Explorer, you enter each expression in a row. Each row has the following columns:

- **Name**

  A variable name for the value computed by the row. This variable is internal to the analysis. You can use this variable as an input to expressions specified by subsequent rows in the analysis.

- **Expression**

  The specification of a calculation performed. The expression can include PI AF attributes, variables from earlier rows in the analysis, and functions.

- **Value at Evaluation**

  The current computed value of the expression. Click **Evaluate** to compute again.

- **Value at Last Trigger**

  The value when the last trigger was computed.

- **Output Attribute**

  For expression analyses, the attribute that stores the computed value. If the analysis contains multiple expressions, you can map the value from any expression to an output attribute. However, at least one expression must be mapped to an output attribute.

For example, an expression analysis might contain one expression, specified in a single row.

Name	Expression	Value at Evaluation	Output Attribute
Variable1	PrevVal('Att1') - 100	505	Analysis1_Output

You can also specify the calculation of an analysis with multiple expressions across several rows. For example, an analysis might contain three expressions: the first two calculate variables V1 and V2, which are used as inputs to the last expression.

Name	Expression
V1	'Att1'/2
V2	'Att2' - 'Att3'
Result	V1 + 3*V2

Analyses with multiple expressions evaluate the expressions in order: the top row first, the one below it next, and so on. Because expressions in lower rows may depend on results from higher rows, you can reorder the rows to evaluate them in the proper order.

You can click **Add a new variable** to add a row and the ⊗ icon to delete it.

Tip:

- You can add a line break within the expression by clicking shift+enter.
- You can make your expressions easier to read by adding comments next to or within an expression. Use double slash (//) to introduce a comment or place the comment between slash asterisk (/*) and asterisk slash (*/).

```
// Add comments here
IF x THEN y ELSE /*Do this if x is false*/ z
```

Name	Expression
Variable1	TimeEq('sinusoid', '*-24h' /* Indicates start time*/, '*' /*Indicates end time*/, 80) //Find the total time within the past 24 hours when the value of 'sinusoid' was 80)

Topics in this section

- Expression specification
- Expression simplification with refactoring

Expression specification

To create an expression, click in the **Expression** column of a row and then specify the input attributes, variables, and functions that define a calculation. You can do the following:

- Select and insert attributes and functions into the expression from the **Functions** and **Attributes** and panels to the right of the expression.
- Enter the text directly into the expression. As you type, functions, attributes, and variables that match what you entered appear in a selection list at the cursor: you can select an item you want to insert.

  Attributes and time expressions are enclosed in single quotes.

  Strings are enclosed in double quotes.

Expression simplification with refactoring

In real-world applications, the expression in an analysis can be extremely complex; lengthy attribute names can add to the complexity. To simplify a complicated expression, refactor it into a group of smaller expressions assigned to variables.

Refactored analyses are easier to understand. Refactored analyses are also easier to test and debug: you can evaluate the smaller expressions one at a time and isolate any errors.

To refactor an analysis, define components of the analysis as separate expressions and reference by variable name. For example, you might define a separate expression that defines a long attribute name as a separate variable or that defines a particular calculation as a separate variable. To create these separate expressions, simply add new expression rows to the analysis and enter text in the **Name** and **Expression** columns.

To reduce typing, you can select a term in an expression, right-click the selection, and then click **Assign to variable**. In the row above the expression, PI System Explorer inserts a new row that maps the selected term to a new variable name. In the original expression, PI System Explorer replaces the term with the new variable name.

You can edit the name of any row. However, if you change the name of an expression row already used as a variable in other expressions, you break the connection between the expressions.

Example

Suppose you have a complex expression defined in a single row:

Name	Expression
`Variable1`	`2*'LongAttributeName' + Avg('Att2', 'Att3', 'Att4')`

You can refactor the expression into the following three rows that contain simpler expressions. Note that third row uses the names of the earlier rows as variables.

Name	Expression
`Variable1`	`'LongAttributeName'`
`Variable2`	`Avg('Att2', 'Att3', 'Att4')`
`Variable3`	`2*Variable1 + Variable2`

Future data and analyses

You can use future data as input to an analysis. You can also use an analysis to produce future data by specifying a future time stamp for the output from an analysis.

Expressions can use values from future events as input to an analysis. However, if an analysis uses event-triggered scheduling, inputs with future data can generate analysis evaluations. If a trigger attribute has future events, an evaluation occurs as each event becomes current, when the clock time coincides with the time of that event.

The following sections show two uses of future data in analyses. The first example shows how to use future data as an input to an analysis. The second example shows how to use an analysis to produce future data.

Example: Input future data to evaluate forecast values

Suppose you need to evaluate forecast values for the outside temperature provided by a third-party service. To do that, create an analysis that finds the difference between forecast and actual values.

The analysis needs two input attributes:

- **ActualTemp**

  Stores near real-time temperature readings from an outside thermometer.

- **ForecastTemp**

  Stores future data, in this case forecast values for the outside temperature each day at 6 a.m., 2 p.m., and 10 p.m.

The analysis calculates the difference between the `ActualTemp` and `ForecastTemp` values and writes results to an output attribute named `DeltaTemp`.

Choose event-triggered scheduling for the analysis with the `ForecastTemp` attribute as the only triggering input attribute. With event-triggered scheduling, an evaluation occurs when the analysis detects a new event for a trigger attribute. Because this trigger attribute stores future data, the analysis detects a new event three times a day, when the clock time reaches 6 a.m., 2 p.m., and 10 p.m.; this triggers an evaluation and a new result for the `DeltaTemp` attribute.

Example: Output future data to calculate weekly emissions goals

Suppose you need to set weekly goals for emissions at a cement plant that operates under a voluntary annual CO_2 cap. The weekly goal will likely change every week in response to the actual emission levels of the past week and to the amount remaining under the voluntary annual cap.

At the end of every week, you know actual emissions since the beginning of the year and can calculate a new weekly emissions goal. You want the time stamp for the new goal to be exactly one week from the calculation time, which coincides with the calculation time of the actual emissions for the next week.

In PI AF, create attributes for the parameters the calculations require, such as the annual cap.

Next, create a new expression analysis that calculates an output attribute named `WeeklyGoal`. The analysis completes the following steps:

1. Find the actual emissions since the beginning of the year.
2. Find the difference between the year-to-date total and the annual cap.
3. Divide the difference by the number of days remaining under the annual cap and multiply by seven to calculate the `WeeklyGoal` attribute.
4. Save the `WeeklyGoal` attribute as a future PI point, which preserves history and allows for trending and other uses.

When creating the analysis, choose periodic scheduling for every day at 12:00 a.m. The syntax of the analysis ensures the calculations only run on Sunday.

Finally, specify the time stamp for the analysis outputs. Enter the expression `* + 7d` to set the time stamp to exactly seven days from the trigger time. The `WeeklyGoal` attribute will have a time stamp that is one week in the future (that is, the following Sunday at 12:00 a.m.).

Expression analyses

Expression analyses calculate one or more output values from specified functions, operators, and input values. You specify the inputs and calculations for these analyses in expressions. You can map any output that an expression calculates to an attribute. You must map at least one output to an attribute.

You can map the output to an existing attribute or you can create a new attribute when you map the output. If you create new attribute when you map the output (from the **Analyses** tab in PI System Explorer), you must choose whether or not to save the output history; the system creates the appropriate type of attribute:

- If you choose to save output history, the system creates an attribute with a PI point data reference and creates a PI point in the specified PI Data Archive. With the output saved in a PI point, you can use visualization tools to see the trend associated with the calculation, and you can retrieve the value for any time that the analysis performed the calculation. The analysis calculates and stores a value as specified by the analysis schedule.
- If you choose not to save output history, PI System Explorer creates an attribute with an analysis data reference. An analysis only calculates the value for expressions mapped to attributes with an analysis data reference when requested. For best performance, map output attributes to attributes with analysis data references if you only need to know the most recent value and do not need previous values.

Topics in this section

- Create an expression analysis template
- Create an expression analysis

Create an expression analysis template

In PI System Explorer, you can create an expression analysis template for an element template. In the template, you can create an expression for each calculation. To record the value that an expression calculates, map the output to an attribute template. You can create new attribute templates from the expression-analysis template.

Before you start

Determine output values you want to calculate and save. Identify the functions, operators, and input attributes needed for the calculations.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Expression**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.
7. Optional. To temporarily evaluate expressions in the analysis template based on values of a particular element, click the **Select an example element** link and select an element based on the current element template. If no elements have been created from the current element template, you must create an element first.

   Templates have no concrete data associated with their attributes. You cannot evaluate an expression in an analysis template, unless you borrow attribute values from a specific element.

   PI System Explorer sets **Example Element** to the element you selected.
8. Specify one or more expressions for the analysis.

   Enter each expression in a row. An expression specifies inputs and calculations. See Expressions for analyses.
9. Map at least one expression to an output attribute template.

a. In the **Output Attribute** column for the expression, click **Map**.

b. Specify the attribute template.

To ...	Do this ...
Map the output to an existing attribute template	Click the name of the attribute template. Select an attribute template with a PI point data reference to store the history of the calculation. For best performance when you do not require history, use an attribute template with an analysis data reference.
Map the output to a new attribute template	a. Click **New Attribute Template** to open the Attribute Template Properties window. b. Indicate whether the attribute should save the history of the output: ▪ Click **Yes** to create an attribute with a PI point data reference. This attribute saves the history of the output. ▪ Click **No** to create an attribute with an analysis data reference. This attribute calculates a new value when requested. c. Configure the attribute template to create: ▪ **Name** Name of created attribute. ▪ **Description** Optional text that describes the attribute. ▪ **Data Server** For attributes that save output history, the PI Data Archive server that stores the PI point data reference. By default, this is set to `%Server%`, which sets to the default PI Data Archive server. ▪ **Value Type** The data type the attribute stores. After creating the attribute template, you can refine its definition from the **Attribute Templates** tab. d. Click **OK** to create the attribute template.

10. Optional. If you specified an example element, click **Evaluate** to verify that output values of individual expressions.

11. Specify scheduling for the analysis. See Analysis scheduling.

12. Optional. If you specified an example element, verify the analysis by reviewing the results produced with historical data:

    a. In the analysis list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

13. To apply changes and save your work locally, click ✔ on the toolbar.
14. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an expression analysis

In PI System Explorer, you can create a unique expression analysis for an individual element. In the analysis, you can create an expression for each calculation. To record the value that an expression calculates, map the output to an attribute. You can create new attributes as you configure an analysis.

Before you start

Determine output values you want to calculate and save. Identify the functions, operators, and input attributes needed for the calculations.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Expression**.
6. Specify one or more expressions for the analysis.

   Enter each expression in a row. An expression specifies inputs and calculations. See Expressions for analyses.
7. Map at least one expression to an output attribute:

   a. In the **Output Attribute** column for the expression, click **Map**.

   b. Specify the attribute.

To ...	Do this ...
Map the output to an existing attribute	Click the name of the attribute. Select an attribute with a PI point data reference to store the history of the calculation. For best performance when you do not require history, use an attribute with an analysis data reference.
Map the output to a new attribute	a. Click **New Attribute** to open the Attribute Properties window. b. Indicate whether the attribute should save the history of the output: ▪ Click **Yes** to create an attribute with a PI point data reference. This attribute saves the history of the output. ▪ Click **No** to create an attribute with an analysis data reference. This attribute calculates a new value when requested. c. Configure the attribute: ▪ **Name** Name of created attribute. ▪ **Description** Optional text that describes the attribute. ▪ **Data Server** For attributes that save output history, the PI Data Archive server that stores the PI point data reference. By default, this is set to `%Server%`, which sets to the default PI Data Archive server. ▪ **Value Type** The data type the attribute stores. After creating the attribute, you can refine its definition from the **Attributes** tab. d. Click **OK** to create the attribute.

8. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times.
9. Specify scheduling for the analysis. See Analysis scheduling.
10. Optional. To verify the analysis, review the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
11. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.
12. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Rollup analyses

A rollup analysis calculates statistical values such as sum and average for selected attributes associated with an element. For example, a rollup analysis on a factory element might use temperature attribute values for all pumps in the factory to calculate their average temperature.

You can choose either of two sources for attributes to include in a rollup analysis:

- **Attributes of the element**

  For example, you want to verify that the level of a tank is constant by checking that all inflows and outflow sum to zero. From the list of the tank element's attributes, select its inflow and outflow attributes for a summation calculation. A rollup using an element's own attributes is also known as an aggregation.

- **Attributes of the element's child elements**

  For example, you want to calculate average energy consumption for a group of pumps in a refinery. To do that, you create a roll-up analysis on the parent element (the refinery) that averages energy consumption attributes from its child elements (pumps).

Use selection criteria to select attributes for rollup. You can select attributes that match attribute name text you enter or that are associated with a category or element template you specify.

With PI Server 2015 R2 and later versions, you can select child attributes for rollup analyses. For more information, see Create a rollup analysis.

You must specify one or more functions to calculate statistics on the selected attributes in the rollup (see Rollup analysis functions). Be sure each output is mapped to an output attribute.

Be aware that:

- You should select attributes carefully to obtain sensible results. Avoid selecting a set with mixed units of measure (UOM; such as temperature and mass) or value types (such as numeric values and time stamps). As long as the inputs and outputs have different units within the same UOM class, the system makes reasonable effort to convert the input units to that of the output for calculation.
- You can add or delete elements or attributes in your hierarchy without the need to update roll-up analyses. Because a rollup identifies input attributes each time it is executed, it automatically includes any new attributes that meet its selection criteria.
- Some selected attributes (such as those with string values) will not participate in rollup calculations, which operate only on numeric or `DateTime` values.

Topics in this section

- Rollup analysis functions
- Create a rollup analysis template
- Create a rollup analysis

Rollup analysis functions

You can specify one or more of the functions listed below to calculate statistics on the input attributes in a rollup analysis.

> **Note:**
> Input value types numeric, `DateTime` and `Enumeration value` are supported for **`Minimum`**, **`Maximum`**, and **`Count`**. **`Average`** takes numeric and `DateTime` value types. Input values for all other functions must be numeric value type.

Function	Description	Supported data types
Sum	The sum of values from selected attributes.	Numeric
Average	The average (mean) of values from two or more selected attributes.	Numeric, `DateTime`
Minimum	The minimum value from selected attributes.	Numeric, `DateTime`, `Enumeration value`
Maximum	The maximum value from selected attributes.	Numeric, `DateTime`, `Enumeration value`
Count	The number of attributes actually used in rollup calculations, which can differ from the total number of selected attributes. For example, an attribute containing a string value (such as a part name) might meet the selection criteria but would not participate in rollup calculations and would be excluded for **`Count`**.	Numeric, `DateTime`, `Enumeration value`
Median	The middle value of three or more values from selected attributes; for an even-numbered group of values, the average of the two middle values. An equal number of values fall above and below the median.	Numeric
Population standard deviation	The population standard deviation, which is calculated using the entire population. Useful when all values in a population are available to include as inputs for the calculation.	Numeric
Sample standard deviation	The sample standard deviation, which is calculated using values from a sample of a population. Useful to provide an estimate of the population standard deviation when it is impractical or impossible to include all population values in the calculation.	Numeric

Create a rollup analysis template

In PI System Explorer, you can create a rollup analysis template for a selected element template. You select input values for the rollup either from attributes of the selected element template or from the attributes of a child element template (whose parent element is derived from the element template). You can choose an example child element and use its attributes to help you develop a rollup analysis template.

Before you start

Identify the statistical functions you want the rollup to calculate and where you want to save the results. You can map results from a function to an existing attribute of the element template, or you can create a new output attribute when you configure an analysis template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Rollup**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.
7. Under **Rollup attributes from**, indicate the source of input values:
   - Click **Child elements of *Element Template Name*** to roll up attributes of child elements of an element based on the current element template.
   - Click **This element - *Element Template Name*** to roll up attributes of an element based on the current element template.
8. Select an example element to provide candidate attributes:
   a. Click the **Select an example element** link to open a window that shows elements derived from the selected element template.
   b. Select an element and click **OK**.
9. Select the attributes to include in the rollup:

If rollup attributes come from:	Then:
Child elements	a. From the **Sample Child Element** list, select the element from which you want to view attributes. The table shows attributes of the selected element. To improve performance, the table only shows a limited number of attributes. You can click the **Show more attributes** link below the table to show additional attributes from the selected element. b. To the left of the table, specify criteria that select attributes to roll up: ▪ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp", such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ▪ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ▪ **Attribute Category** Select a category to limit selected attributes to those in that category. ▪ **Element Category** Select an element category to limit selected attributes to attributes that have parent elements in that category. ▪ **Element Template** Select an element template to limit selected attributes to attributes that have parent elements based on that template. **Note:** The criteria apply to all attributes of any child element, not just those shown in the table. You can specify criteria that select attributes not shown in the table. For example, if you set **Sample Child Element** to an element from category `CatX` but set **Element Category** to `CatY`, you will select none of the attributes in the table but the analysis will still include the attributes in the rollup.

This element	To the left of the table, specify criteria that select attributes to roll up: ◦ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp", such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ◦ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ◦ **Attribute Category** Select a category to limit selected attributes to that category.

The table shows a check mark next to any displayed attribute included in the rollup.

10. In the functions table, specify the rollup calculations and output:
    a. Select the check boxes next to any statistical function you want the rollup to calculate.
    b. Click **Map** to store results from a calculation to an output attribute.

       You can map the output to an existing attribute or map the output to a new output attribute.
    c. Optional. At the top of the table, click **Evaluate** to verify the output values of the selected functions.
11. Specify scheduling for the analysis. See Analysis scheduling.
12. Optional. To verify the analysis, examine the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
13. To apply changes and save your work locally, click ✔ on the toolbar.
14. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create a rollup analysis

In PI System Explorer, you create a rollup analysis for a selected element. You select input attributes for the rollup either from attributes of the selected element or from attributes of child elements.

Before you start

Identify the statistical functions you want the rollup to calculate and where you want to save the results. You can map results from a function to an existing attribute of the element, or you can create a new output attribute when you configure the analysis.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Rollup**
6. Under **Rollup attributes from**, indicate the source of input values:
   - Click **Child elements of *Element Name*** to roll up attributes of a child element.
   - Click **This element - *Element Name*** to roll up attributes of the current element.
7. Select the attributes to include in the rollup:

If rollup attributes come from ...	Do this ...
Child elements	a. From the **Sample Child Element** list, select the element from which you want to view attributes. The table shows attributes of the selected element. To improve performance, the table only shows a limited number of attributes. You can click the **Show more attributes** link below the table to show additional attributes from the selected element. b. To the left of the table, specify criteria that select attributes to roll up: ▪ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp," such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ▪ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ▪ **Attribute Category** Select a category to limit selected attributes to those in that category. ▪ **Element Category** Select an element category to limit selected attributes to attributes that have parent elements in that category. ▪ **Element Template** Select an element template to limit selected attributes to attributes that have parent elements based on that template. **Note:** The criteria apply to all attributes of the selected child element, not just those shown in the table. You can specify criteria that select attributes not shown in the table. For example, if you set **Sample Child Element** to an element from category `CatX` but set **Element Category** to `CatY`, you will select none of the attributes in the table but the analysis will still include the attributes in the rollup.

This element	To the left of the table, specify criteria that select attributes to roll up:
	◦ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp," such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ◦ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ◦ **Attribute Category** Select a category to limit selected attributes to that category.

The table shows a check mark next to any displayed attribute included in the rollup.

8. In the functions table, specify the rollup calculations and output:
   a. Select the check boxes next to any statistical function you want the rollup to calculate.
   b. Click **Map** to store results from a calculation to an output attribute.

      You can map the output to an existing attribute or map the output to a new output attribute.
   c. Optional. At the top of the table, click **Evaluate** to verify the output values of the selected functions.
9. Specify scheduling for the analysis. See Analysis scheduling.
10. Optional. To verify the analysis, review the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
11. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.
12. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Event frame generation analyses

An event frame generation analysis specifies the conditions to start and end event frames automatically.

This type of analysis includes either one or two expressions. When a single condition triggers both the start and the end of an event frame, only a start trigger expression is needed, depicted by the **StartTrigger1** field. For example, a temperature value rising above a threshold might start an event frame and end it when the value returns below the threshold. When the start and end conditions are different, an **EndTrigger** expression is also needed. Because they test start and end conditions, expressions in event frame generation analyses must evaluate to true or false. For more information, see Start and end trigger conditions.

You can write up expressions to be used as start or end triggers. If your expression gets very complex, you can break it into a group of smaller expressions and assign them to variables. For more information, see Expression simplification with refactoring.

Beginning with the 2018 release, a new way of event frame generation is possible where you do not have to set an explicit trigger. For more information, see Implicit modes of event frame generation in asset analytics.

When configuring event frame generation analyses, you can add expressions to **Outputs at Close** group to write outputs back to event frame attributes. Depending on whether you want to save the history of the output, you can select either attributes configured with a PI point data reference or static event frame attributes. When saving output history to event frame attributes configured as a PI Point data reference, the attribute must be mapped to a corresponding PI Point data reference attribute of the referenced element.

Note:

- Outputs from root cause event frames are only written to static attributes.
- Parent event frames with multiple child event frames will also write only to static attributes. For more information on child event frame generation, see Child event frame generation with multiple start triggers.

In your expressions, you can use inputs that come from assets to write outputs to event frame attributes at the close of an event frame. You can also use the **`EventFrame`** function to access the start time, end time and duration of the event frame. For more information, see expression function EventFrame.

An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

Note:

- Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
- An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
- Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff%`) will be used in the absence of a name or if the event frame name evaluates to a null value.

You can create multiple start triggers for your analysis and specify different Boolean conditions to trigger each event frame. For more information on multiple start triggers and child event frame generation, see Child event frame generation with multiple start triggers.

> **Note:**
> If you change the name of a start trigger from the default name, you must change all the start triggers and use unique names for each trigger.

A spike in input data can trigger the start of an unwanted event frame. To counter the effects of data spikes, you can require that the start trigger remain true for a set time interval before creating an event frame. Enter a value in the **True for** field to specify the time interval.

You can also specify a Severity level for the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical.

An event frame generation analysis is based on an event frame template, which specifies the attributes for the generated event frames. Before you create an event frame generation analysis, be sure an appropriate event frame template is available.

Event frames usually include a referenced element collection. The element associated with an event frame generation analysis becomes the primary reference element for its generated event frames. For more information, see Primary referenced element. Other elements can be referenced using relative paths based on the primary referenced element, as described in Data references to attributes in other elements.

For some events, such as a forced shutdown, you may want to evaluate the conditions leading up to the event. To do that, you can have PI Analysis Service generate root cause event frames. When configured, every generated event frame includes a child root cause event frame that captures attributes for a specified time interval just before the start of the generated event frame. For more information on generating a root cause event frame, see step 11 on Create an event frame generation analysis template.

Video

For information on how to set up event frame generation analyses, watch this video:

https://www.youtube.com/watch?v=iNYe_YZzj58&rel=0

Topics in this section

- Implicit modes of event frame generation in asset analytics
- Create an event frame generation analysis template
- Create an event frame generation analysis
- Start and end trigger conditions
- Child event frame generation with multiple start triggers

Implicit modes of event frame generation in asset analytics

Beginning with the 2018 release, users of asset analytics in PI AF have the option to select different modes of event frame generation. Prior to 2018, users had to explicitly set a start trigger (now called Explicit Trigger). In 2018 or later versions of PI AF, you can choose among the three implicit modes in addition to an explicit trigger: pulse, step and step continuous. After having selected Event Frame Generation as your analysis type, select one of the following as **Generation Mode**:

- Pulse: A transition from zeroth state to any other state starts an event frame, and transition to the zeroth state ends the event frame. You have an option to generate a root cause event frame in **Advanced Event Frame Settings**.
- Step: All transitions would close the open event frame and start a new event frame unless the transition is to a zeroth state
- Step Continuous: All transitions would close the open event frame and start a new event frame

Note that you must select a triggering attribute that is either integer, string or enumeration value for these event frames. For pulse and step event frames, you will need to select an attribute as a trigger and set zeroth state. Zeroth state is not supported for step continuous event frames. Zeroth state determines when the event frame should be closed. You can select zeroth states from digital state or enumeration sets. For more information, see PI EFGen topic "Event start and end detection" in Live Library (https://livelibrary.osisoft.com).

Create an event frame generation analysis template

In PI System Explorer, you can create an event frame generation analysis template for an element template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Event Frame Generation**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.

7. Optional. To temporarily evaluate expressions in the analysis template based on values of a particular element, click the **Select an example element** link and select an element based on the current element template. If no elements have been created from the current element template, you must create an element first.

   Templates have no concrete data associated with their attributes. You cannot evaluate an expression in an analysis template, unless you borrow attribute values from a specific element.

   PI System Explorer sets **Example Element** to the element you selected.

8. From the **Event Frame Template** list, select the template for the event frame that this analysis generates.

   To create an event frame template, see Create event frame templates. An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

   Note:

   - Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
   - An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
   - Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff %`) will be used in the absence of a name or if the event frame name evaluates to a null value.

9. Specify the necessary expressions for the analysis:

   a. In the **StartTrigger1** row, enter the Boolean expression that specifies the condition that starts an event frame.

   b. Optional. Change the default name of the start trigger.

      Note:
      If you change the name of a start trigger from the default name, you must change the names of all start triggers and use unique names for each trigger.

   c. Optional. In the **True for** field, enter the amount of time that the start-trigger condition must be true before the analysis starts an event frame.

      Note:
      Specify a value in the **True for** field to reduce unwanted event frames caused by momentary fluctuations in input data.

   d. Optional. In the **Severity** field, enter the severity of the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical. If you have defined multiple start triggers, the one with the highest severity will be the effective start trigger; if all start triggers have the same severity, the first start trigger in the list will be the effective start trigger.

   e. Add multiple start triggers by clicking on **Add a new start trigger**. For each start trigger, enter a Boolean expression, specify a value for **True for** and select a value for **Severity**.

> **Note:**
> If you have changed the default name of a start trigger, make sure to assign unique names to all the start triggers.

f. Optional. If a different condition ends the event frame, enter the expression in the **EndTrigger** row. Otherwise, the event frame will close when the start trigger condition is no longer true.

g. Optional. Click **Add a new variable** to insert a new expression row where you can specify a variable and expression for use in one of the trigger expressions.

h. Optional. Click **Add a new output expression** to insert a new expression to write outputs to event frame attributes. You can map the output as static (doesn't save history) or PI point (saves history) attribute. When mapping to PI point attribute, make sure that it is mapped back to the attribute on the asset.

10. Optional. If you specified an example element, click **Evaluate** to verify that output values of individual expressions.

11. Optional. Generate a root cause event frame for every event frame that the analysis generates. A root cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

> **Note:**
> Outputs from root cause event frames are only written to static attributes.

a. From the Advanced Event Frame Settings window, select **Generate child root cause event frame before parent event frame starts**.

b. In the **Duration** field, enter the length of the root cause event frame.

The root cause event frame starts this long before the generated parent event.

12. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger name is saved.

b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger expression is saved.

The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

13. Specify scheduling for the analysis. See Analysis scheduling.

14. Optional. If you specified an example element, verify the analysis by reviewing the results produced with historical data:

a. In the analysis list, right-click the analysis and then click **Preview Results**.

b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

15. To apply changes and save your work locally, click ✔ on the toolbar.

16. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an event frame generation analysis

In PI System Explorer, you can create an event frame generation analysis for an individual element.

Before you start

Check that an event-frame template is available for the analysis.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Event Frame Generation**.
6. From the **Event Frame Template** list, select the template for the event frame that this analysis generates.

   To create an event frame template, see Create event frame templates. An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

Note:

- Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
- An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
- Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff %`) will be used in the absence of a name or if the event frame name evaluates to a null value.

7. Specify the necessary expressions for the analysis:
   a. In the **StartTrigger1** row, enter the Boolean expression that specifies the condition that starts an event frame.
   b. Optional. Change the default name of the start trigger.

      Note:
      If you change the name of a start trigger from the default name, you must change all the start triggers and use unique names for each trigger.

   c. Optional. In the **True for** field, enter the amount of time that the start-trigger condition must be true before the analysis starts an event frame.

      Note:
      Specify a value in the **True for** field to reduce unwanted event frames caused by momentary fluctuations in input data.

   d. Optional. In the **Severity** field, enter the severity of the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical. If you have defined multiple start triggers, the one with the highest severity will be the effective start trigger; if all start triggers have the same severity, the first start trigger in the list will be the effective start trigger.
   e. Add multiple start triggers by clicking on **Add a new start trigger**. For each start trigger, enter a Boolean expression, specify a value for **True for** and select a value for **Severity**.

      Note:
      If you have changed the default name of a start trigger, make sure to assign unique names to all the start triggers.

   f. Optional. If a different condition ends the event frame, enter the expression in the **EndTrigger** row. Otherwise, the event frame will close when the start trigger condition is no longer true.
   g. Optional. Click **Add a new variable** to insert a new expression row where you can specify a variable and expression for use in one of the trigger expressions.
   h. Optional. Click **Add a new output expression** to insert a new expression to write outputs to event frame attributes. You can map the output as static (doesn't save history) or PI point (saves history) attribute. When mapping to PI point attribute, make sure that it is mapped back to the attribute on the asset.

   See Expressions for analyses for more information.

8. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times.

9. Optional. Generate a root cause event frame for every event frame that the analysis generates. A root cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

   Note:
   Outputs from root cause event frames are only written to static attributes.

   a. From the Advanced Event Frame Settings window, select **Generate child root cause event frame before parent event frame starts** .

   b. In the **Duration** field, enter the length of the root cause event frame.

      The root cause event frame starts this long before the generated parent event.

      Note:
      The root cause event frame is created only for the first instance, not for each time that the trigger is `True`.

10. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

    Note:
    We recommend that you provide meaningful names to your analysis start triggers while saving them to an event frame attribute.

    a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger name is saved.

    b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger expression is saved.

       The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

11. Specify scheduling for the analysis. See Analysis scheduling.

12. Optional. To verify the analysis, review the results produced with historical data:

    a. In the analysis list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

13. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.

14. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Start and end trigger conditions

An event frame generation analysis starts an event frame when the start trigger condition evaluates to true. Because a random spike in an input value can make the condition true, noisy data can lead to the creation of many unwanted event frames.

To counter the effects of data spikes, you can require that the start trigger condition remain true for a set time interval before starting an event frame. Enter a value in the **True for** field to specify that time interval.

The event frame generation analysis will not create a new event frame until the expression in the start trigger has been true for the duration specified in **True for** field. The start time for the event frame is the time that the start trigger condition first evaluated to true.

If the start trigger condition evaluates to false before it reaches the time interval specified in **True for** field, no event frame is created.

The **True for** value has no effect on event-frame end time.

The event frame will close when the start trigger condition is false. Configuring an end trigger is optional. If both the start and end trigger conditions are configured, the event frame will end when the end trigger condition is met. This is regardless of whether the start trigger condition still holds true.

Tip:

- If using both start and end triggers, make sure the expressions never evaluate to TRUE at the same time since this may lead to event frames with zero second durations.
- Try to configure your event frames to use only a start trigger expression.

Child event frame generation with multiple start triggers

PI Analysis Service can capture child event frames. Child event frames are useful when you need to send escalating notifications or perform an in-depth analysis of the past events. This topic discusses the following:

- Video that shows you how to configure event frames with multiple start triggers
- Effective start triggers
- Example of child event frame generation

Note:
This topic is only applicable to the explicit trigger.

Video

For information on setting up event frames with multiple start triggers and severities, watch this video:

https://www.youtube.com/watch?v=R15X4fYYxAQ&rel=0

Effective start triggers

Event frame generation analysis supports multiple start triggers. When triggering an analysis, the *effective start trigger* is determined by taking the following three things into account: trigger evaluation, severity and order of the triggers.

- **Evaluation**

  If only one start trigger is `True`, then that is the effective start trigger. If multiple start triggers evaluate to `True`, then the analysis must take severity into account.

- **Severity**

  If the multiple start triggers evaluate to `True`, the one with the highest severity level is the effective start trigger. For example, if there are two start triggers, `StartTrigger1` with severity `Major` and `StartTrigger2` with severity `Critical`, and they both evaluate to `True` at the same point in time, then `StartTrigger2` is considered the effective trigger as it has the higher severity.

  > **Note:**
  > Severity levels in decreasing order of magnitude are: `Critical`, `Major`, `Minor`, `Warning`, `Information` and `None` (default).

- **Order**

  If multiple start triggers with the same severity evaluate to `True`, then the start trigger defined first is the effective start trigger.

In the end, there is only one effective start trigger.

Child event frame generation due to change in the effective start trigger

An analysis with multiple start triggers may automatically create child event frames. For example, an event frame generation analysis would complete the following steps:

1. A start trigger evaluates to `True` and there is no open event frame. The analysis starts an event frame.
2. In a subsequent evaluation, while the event frame is still open, the effective start trigger does not change and continues to evaluate to `True`. Then the event frame will remain open.
3. Another evaluation causes the effective start trigger to be different than step 1. This causes the analysis to do the following:
   a. Start a new parent event frame with the same start time as the original event frame.
   b. Make the original event frame a child of this new parent event frame.
   c. Close the original event frame (now a child event frame).
   d. Create a new child event frame with the new effective start trigger.
4. Another evaluation does not cause the effective start trigger to change. The analysis keeps both the child and the parent event frames open.
5. Another evaluation causes the effective start trigger to change again. Then:
   a. The child event frame created in 3d closes.
   b. Yet another child event frame is created by this new effective start trigger.
6. Another evaluation causes all of the start triggers to be `False`. Then:
   a. Child event frame created in 5b closes.
   b. Parent event frame created in 3a closes.

SQC analyses

Using PI System Explorer, you can create SQC analyses that use standard Statistical Quality Control (SQC) tests to determine if the value of a PI AF attribute lies within user-specified

margins of error. Further, you can specify either attributes or event frames as outputs of your SQC analysis.

The following SQC pattern tests may be selected:

- **Outside Control:**

  Counts the number of samples outside the control limit on one side of the center line.

- **Outside 2 Sigma:**

  Evaluates the sample against a limit drawn 2/3 of the way between the center line and the control limit.

- **Outside 1 Sigma:**

  Evaluates the sample against a limit drawn 1/3 of the way between the center line and the control limit.

- **One Side of Center Line:**

  Counts the number of samples on one side of the center line.

- **Stratification:**

  Counts the number of samples that fall within the upper and lower One Sigma limits on both sides of the center line.

- **Mixture:**

  Counts the number of samples that fall outside the upper and the lower One Sigma limits on both sides of the center line.

- **Trend:**

  Counts the number of samples which are increasing or decreasing in a monotonic sequence.

Create an SQC analysis template

In PI System Explorer, you can create an SQC analysis template for an element template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.

- **Description**

  Optional text that describes the analysis.

- **Categories**

  Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.

- **Analysis Type**

  Click **SQC**.

6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.

7. Optional. Select an example element.

8. Provide the inputs for the SQC pattern test. All these inputs should be attribute templates previously defined in your PI System.

   a. In the **Source** field, select the PI AF attribute template that provides the data for the pattern test.

   b. In the **Upper Control Limit** and **Lower Control Limit** fields, select the PI AF attribute template that provides the data for the largest and smallest limits on the dimension of the data being tested.

   c. In the **Center Line** field, select the PI AF attribute template that provides the normal or ideal value of the data.

9. Choose the output of your analysis as either **Event Frame** or **AF attribute**. If you choose **Event Frame**, select an event frame template. If you choose **AF attribute**, you can either choose from a list, or create a new attribute template. SQC Analysis rule automatically creates an enumeration set named **SQC Enumeration**. This enumeration set has the same values used in PI Data Archive SQC DigitalState set so that continuity with existing PI Data Archive is preserved. **SQC Enumeration** is created when: 1) you map the output to a new **AF attribute**, 2) this or another enumeration set with the required name and values does not already exist and 3) you have the right to create enumeration sets. **AF attribute** output defaults to a string attribute if the aforementioned conditions are not met.

10. Optional. Generate a root-cause event frame for every event frame that the analysis generates. A root-cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

    a. From the **Advanced Event Frame Settings** menu, select **Generate Child Root Cause Event Frame Before Parent Event Frame Starts** .

    b. In the **Duration** field, enter the length of the root-cause event frame.

       The root-cause event frame starts this long before the generated parent event.

       > **Note:**
       > The root-cause event frame is created only for the first instance, not for each time that the trigger is `True`.

11. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute template on the event frame template to which the start trigger name is saved.

b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute template on the event frame template to which the start trigger expression is saved.

The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

12. Enter SQC pattern test information.

a. Select the check box for each SQC pattern test that you wish to apply to your data.

b. In the **X of Y** column, select appropriate numerical values for X and Y for each selected pattern test that requires this input. Specify the number of values X out of a number of sampled values Y that can be above or below the corresponding pattern test limits.

Patterns (X and Y values) suggested by Western Electric are shown by default.

> **Note:**
> X may not be greater than Y.

c. Select the **Limit** as **Above**, **Below** or **Both**, for each selected pattern test. By default, tests are applied on both sides of the center line.

13. Optional. Select the **Clear on Control Limit Change** check box.

With **Clear on Control Limit Change** selected, if any of the control limits changes, asset analytics resets the state of a running SQC calculation, and the SQC calculation is restarted with the changed values. In addition, associated open event frames are closed before the calculation is resumed.

With **Clear on Control Limit Change** not selected, the calculation simply proceeds using the changed values for the control limits.

14. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times. If all selected SQC tests pass, the **Output Value** field displays "Normal". If some tests fail, the **Output Value** field shows the highest priority test that failed. Changing any of the SQC inputs will clear the **Output Value** field.

15. Specify scheduling for the analysis. See Analysis scheduling.

16. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an SQC analysis

Before you start

Identify the Statistical Quality Control (SQC) pattern tests that you wish to use. See SQC analyses for more information.

Procedure

1. In the navigator, click **Elements**.

2. In the Elements browser, select the element where you want to create the analysis.

3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.

4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.

5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **SQC**.

6. Provide the inputs for the SQC pattern test. All these inputs should be attributes or attribute templates previously defined in your PI System.
   a. In the **Source** field, select the PI AF attribute or attribute template that provides the data for the pattern test.
   b. In the **Upper Control Limit** and **Lower Control Limit** fields, select the PI AF attribute or attribute template that provides the largest and smallest limits on the dimension of the data being tested.
   c. In the **Center Line** field, select the PI AF attribute or attribute template that provides the normal or ideal value of the data.

7. Choose the output of your analysis as either **Event Frame** or **AF attribute**. If you choose **Event Frame**, select an event frame template. If you choose **AF attribute**, you can either choose from a list, or create a new attribute. SQC Analysis rule automatically creates an enumeration set named **SQC Enumeration**. This enumeration set has the same values used in PI Data Archive SQC DigitalState set so that continuity with existing PI Data Archive is preserved. **SQC Enumeration** is created when: 1) you map the output to a new **AF attribute**, 2) this or another enumeration set with the required name and values does not already exist and 3) you have the right to create enumeration sets. **AF attribute** output defaults to a string attribute if the aforementioned conditions are not met.

8. Optional. Generate a root-cause event frame for every event frame that the analysis generates. A root-cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.
   a. From the **Advanced Event Frame Settings** menu, select **Generate Child Root Cause Event Frame Before Parent Event Frame Starts** .
   b. In the **Duration** field, enter the length of the root-cause event frame.

      The root-cause event frame starts this long before the generated parent event.

> **Note:**
> The root-cause event frame s created only for the first instance, not for each time that the trigger is `True`.

9. Optional. Map a start trigger name and start trigger expression to an event frame attribute.
   a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute or attribute template on the event frame template to which the start trigger name is saved.
   b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute or attribute template on the event frame template to which the start trigger expression is saved.

      The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.
10. Enter SQC pattern test information.
    a. Select the checkbox for each SQC pattern test that you wish to apply to your data.
    b. In the **X of Y** column, select appropriate numerical values for X and Y for each selected pattern test that requires this input. Specify the number of values X out of a number of sampled values Y that can be above or below the corresponding pattern test limits.

       Patterns (X and Y values) suggested by Western Electric are shown by default.

       > **Note:**
       > X may not be greater than Y

    c. Select the **Limit** as **Above**, **Below** or **Both**, for each selected pattern test. By default, tests are applied on both sides of the center line.
11. Optional. Select the **ClearOnControlLimitChange** checkbox. With **ClearOnControlLimitChange** selected, if any of the control limits changes, asset analytics resets the state of a running SQC calculation, and the SQC calculation is restarted with the changed values. In addition, associated open event frames are closed before the calculation is resumed.

    With **ClearOnControlLimitChange** not selected, the calculation simply proceeds using the changed values for the control limits.
12. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times. If all selected SQC tests pass, the **Output Value** field displays "Normal". If some tests fail, the **Output Value** field shows the highest priority test that failed. Changing any of the SQC inputs will clear the **Output Value** field.
13. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Sample analyses

The sample analyses in this section perform the following calculations in the same PI AF database:

1. Uses an expression type analysis to track the deviation from target efficiency for the asset and processes represented by the element, `Ethylene`.
2. Creates a template to generate a set of rollup analyses that aggregate the total power draw of all processes at a refinery. The element template from which the element for each refinery was created is called `Refinery`.
3. Creates event frames to capture data any time the 15-minute running-average efficiency for the element, `Ethylene`, drops below 90%.

Build an expression analysis to study efficiency deviation

This example demonstrates how to build an expression type analysis. The analysis tracks how the efficiency of a process deviates from the target efficiency.

First, we'll calculate the hourly average efficiency, then determine how much it varies from the target efficiency.

This example creates a new PI point to store the output data (the deviation from target efficiency, as it varies over time). When you create your own analyses, OSIsoft recommends you create the PI points you'll need for output data before you start to create the analysis.

Before you start

Create an element named `Ethylene` that has an attribute named `Efficiency` that contains a PI point data reference.

Procedure

1. Under **Elements**, select the element for which you want to build the analysis, such as **Ethylene**.

2. Click the **Analyses** tab.

3. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
4. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis. For example, `EfficiencyCalc`.
   - **Analysis Type**

     Click **Expression**. The expression analysis is the simplest type of analysis; it contains one or more expressions plus scheduling information. For more information, see Expression analyses.
5. In the expressions table, create a variable to hold the constant value of the target efficiency:
   a. In the **Name** column, type `Target`.

   b. In the **Expression** column, enter `90`.

   This sets this variable to the constant value of 90.
6. Create a second variable to calculate the hourly average efficiency:
   a. Click **Add a new expression**.
   b. Name the variable `HourlyEfficiency`.
   c. In the **Expression** column, enter the following performance equation syntax:
   ```
   TagAvg('Efficiency', '*-1h', '*')
   ```

   This expression calculates the average value of the attribute called `Efficiency` over the past hour.
7. Create a third variable to calculate how much the hourly efficiency deviates from the target of 90:
   a. Click **Add a new expression**.
   b. Name the variable `Deviation`.
   c. In the **Expression** column, enter the following expression:
   ```
   HourlyEfficiency - Target
   ```
8. Save the output from the `Deviation` variable to a PI point:
   a. In the **Output Attribute** column, click **Map**.
   b. Click **New Attribute** to create a new PI AF attribute.
   c. To define the attribute as a PI point data reference, click **Yes** to save output history.
   d. In the **Name** field, enter the attribute name, such as `EfficiencyDifference`.
   e. Click **OK** to create the attribute.

9. Specify scheduling for the analysis:
   a. Set the **Scheduling** option to **Periodic**.
   b. Click **Configure** and specify a period of 30 seconds.
   The calculation will run every 30 seconds.
10. Verify that the results look reasonable:
   a. In the analyses table, right-click the analysis and then click **Preview Results**.
   b. Enter the time range of the preview. The default time range starts at the preceding midnight and ends at the current time, but you can enter different start and end times, if you prefer.
   c. Click **Generate Results**.
11. When you are satisfied with your new analysis, click **Check In** on the toolbar to check in your work.

   PI System Explorer saves and checks in your analysis, and then starts running the analysis.

   A green circle in the **Status** column of the analyses table indicates that PI Analysis Service is running the analysis.
12. Optional. Back fill the PI point with results from the calculation using past data.

   PI Analysis Service can back fill calculation results that have output attributes mapped to PI point data references. However, the PI point cannot contain data at the specified times. If desired, you can delete pre-existing data and then back fill that time range.

   a. Right-click the analysis in the analyses table, and click **Backfill/Recalculate**.
   b. Specify a start and end time for back filling.
   c. Select **Permanently delete existing data and recalculate**.
   d. Click **Start**.

Create a template for power rollup analyses

This example demonstrates how to create a template that you can use to generate a set of rollup analyses. The analysis template aggregates the total power draw of all processes at a refinery.

Before you start

Create the necessary objects in your PI AF database:

- Element template named `Refinery`
- Element named `Rotterdam Refinery`, derived from the `Refinery` template
- Child element named `Catalytic Cracking` under the `Rotterdam Refinery` element
- Attribute named `Power Draw` under the `Catalytic Cracking` child element

Procedure

1. In the Library browser, select the element template for which you want to create the analysis template.

In this example, the element template is called `Refinery`. This template is used to create refinery elements.

2. Click the **Analysis Templates** tab.
3. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
4. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis. For example, `PowerRollup`.
   - **Analysis Type**

     Click **Rollup**. A rollup analysis calculates statistical data for specified attributes, such as a total or an average value. For more information, see Rollup analyses.
5. Select an example element to provide candidate attributes:
   a. Click the **Select an example element** link to open a window that shows elements derived from the selected element template.
   b. Select an element and click **OK**.

      For example, in the list of elements derived from the `Refinery` template, click **Rotterdam Refinery**.
6. For the **Rollup attributes from** option, click **Child elements of Rotterdam Refinery** to indicate that the analysis rolls up attributes of child elements of the refinery element.
7. Select the attributes to include in the rollup:
   a. From the **Sample Child Element** list, select the element from which you want to view attributes.

      For example, select **Catalytic Cracking** to see attributes from the `Catalytic Cracking` element in the attributes table.

      The table only shows attributes from the selected element. The rollup include attributes from all child elements that match the criteria that you specify.
   b. To the left of the attributes table, specify criteria that select attributes to roll up.

      For example, in the **Attribute Name** field, type `Power*` to specify all attributes that begin with `Power`. The attributes table shows a check mark next to any attributes in the selected child element that match this criteria. However, the rollup includes attributes from any child element that match the criteria, not just those in the table.
8. In the functions table, specify the rollup calculations and output:
   a. Select the check boxes next to any statistical function you want the rollup to calculate.

      For example, click the **Sum** check box to total values.
   b. In the **Output(s)** column, click **Map** to store results from the calculation to an output attribute.

c. Click **New Attribute Template** to create a new attribute template with a PI point data reference.

d. Specify values that define the created attribute and point, and then click **OK**.

For example, you can use the default values, which name the point and attribute `PowerRollup_Sum` and write the point to the default PI Data Archive, indicated by the substitution parameter.

9. Click **Evaluate** and check whether the result returned in the **Value** column is reasonable.
10. Set the **Scheduling** option to **Event-Triggered**.
11. To verify the analysis, examine the results produced with historical data:

    a. In the analyses list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

12. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create event frames automatically to track inefficiency

We created a sample analysis to track how the efficiency of a process deviates from the target efficiency in Build an expression analysis to study efficiency deviation. In this example, we create event frames to capture data any time the 15-minute running-average efficiency drops below 90%.

Before you start

Complete the steps in Build an expression analysis to study efficiency deviation.

Procedure

1. In the Library browser, expand **Templates**.
2. Right-click **Event Frame Templates** and then click **New Template** to create a new event frame template.
3. In the **Name** field, enter `EfficiencyAnomaly`.
4. In the **Naming Pattern** field, specify the name of the event frames produced from the template.

   You can use substitution parameters to specify a varying name. PI AF resolves the parameters when creating the event frames. Then, each event frame will have a unique, identifiable name.

   a. Click the arrow next to the field to see valid substitution parameters that you can add to the name.

   b. Select **%ELEMENT%**.

   c. Select **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%**.

   The **`STARTTIME`** substitution parameter includes the date-time formatting pattern to use.

5. Click the **Attribute Templates** tab.
6. Create a new PI point data reference attribute using an attribute template.
   a. Click **New Attribute Template**.
   b. In the **Name** field, enter `Efficiency`.
   c. Click **Settings** and enter the following configuration string in the **Tag name** field:

      `.\Elements[.]|Efficiency; uom=%`

   d. From the **Default UOM** list, select **percent**.

7. In the Elements pane, click **Ethylene**, the element for which you built an analysis in Build an expression analysis to study efficiency deviation.
8. Click the **Analyses** tab.
9. Click **New Analysis** .
10. Enter information to identify the analysis:
    - **Name**

      Enter `EfficiencyAnomalyEvent`.

- **Analysis Type**

  Click **Event Frame Generation**.

11. From the **Event Frame Template** list, select **EfficiencyAnomaly**.

    This is the event frame template you created in the earlier steps.

12. Enter expressions that specify the start and end of the event frame:

    - **StartTrigger1**

      ```
      TagAvg('Efficiency', '*-15m', '*') <90
      ```

      Creates an event frame whenever the 15-minute average efficiency drops below 90%.

    - **EndTrigger**

      ```
      TagAvg('Efficiency', '*-15m', '*') >92
      ```

      Ends the event frame whenever the 15-minute average efficiency has recovered and exceeds 92%.

13. Click **Evaluate** to check the current values of the expressions.

    The **Value** column is populated with either `True` or `False` for each condition.

14. Specify scheduling for the analysis.

    a. For the **Scheduling** option, click **Periodic**.

    b. Click **Configure** and set the analysis to run every minute.

15. Check in your work.

Backfilling and recalculation of analyses

Backfill or recalculate

When you backfill an analysis for a specified time range, data that are missing are calculated to fill in the gaps for the outputs. Recalculation, on the other hand, deletes all of the data in the time range and calculate results for the analysis outputs again. Currently, there are four different options of backfill and recalculation. For more information, see OSIsoft Tech Support Article KB01697 - Asset analytics backfilling and recalculation milestones and requirements (https://techsupport.osisoft.com/Troubleshooting/KB/KB01697/).

Expression, rollup or SQC analyses can be backfilled or recalculated over an earlier time period if the analyses outputs are mapped to PI point attributes. You can backfill the missing data in a time range and retain existing data, or you can recalculate, which deletes and recalculates data in the time range. Event frame generation analyses can recalculate event frames over a specified time period, but automatically delete all existing event frames in that time period, as well as annotations on affected event frames.

In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. For more information on identity mapping, see PI AF identities and mappings.

> **Note:**
> For analyses writing outputs to attributes configured as PI point, backfilling or recalculation may result in writing out-of-order events. Such events are written to the PI Data Archive without compression.

Enabling automatic backfilling is useful as it fills the gaps in case a short analysis service downtime occurs. For more information, see parameters **`AutoBackfillingEnabled`** and **`MaximumAllowedAutoBackfillingSpanInHours`** in Analysis service configuration.

From PI AF 2017 R2, you can enable automatic recalculation of analyses. PI AF Server and PI Analysis Service 2017 R2 or later are required to make use of automatic recalculation. Although automatic recalculation is globally enabled by default (**`AutoRecalculationEnabled`**), you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation. For more information, see Configure automatic recalculation of analyses and analyses templates, Enable or disable automatic recalculation for multiple analyses and parameters **`AutoRecalculationEnabled`**, **`AutoRecalculationIgnoreTimeInSeconds`**, **`AutoRecalculationMinWaitTimeInSeconds`**, and **`MaxAllowedAutoRecalculationSpanInDays`** in Analysis service configuration. To examine a log file with a list of queued automatic and manual backfill/recalculation requests for troubleshooting, see Open recalculation log folder.

Note:
Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Backfill or recalculate data for an analysis

Before you start

PI Analysis Service 2016 R2 introduced recalculation capabilities for expression, rollup and SQC analyses. PI Data Archive storing the output PI points must be running version 2016 R2 or later. Beginning with PI Analysis Service 2018, you can opt for recalculation of all dependent analyses. An analysis whose input comes from an output of another analysis is considered dependent.

Procedure

1. In the Elements browser, select an element, and click the **Analyses** tab to see the analyses defined for the element.
2. From the list of analyses, right-click the running analysis that you want to use to backfill or recalculate data, and click **Backfill/Recalculate**.
3. In the **Backfilling or recalculation for *analysis name*** window, perform the following actions.
   a. In the **Start Time** and **End Time** fields, specify the time period for which you want to backfill missing data while retaining existing data, or delete and recalculate data. You can use standard PI time expressions or click to select a specific date period.
   b. Decide how existing data should be treated. For an expression, rollup or SQC analysis that outputs to an attribute:
      - To ensure that existing data is retained and only missing data is backfilled, select **Leave existing data and fill in gaps**.
      - If input data or analysis algorithms have changed and you want to delete existing output data, select **Permanently delete existing data and recalculate**.
        - Optional. Check the box next to **Recalculate dependent analyses** to queue all dependent analyses for manual recalculation.

Note:

- Selecting this option may cause recalculation to occur multiple times for dependent analyses
- The recalculation time range for dependent analyses may be different than the time range you selected when queuing recalculation. For more information, see note at the end of step 7 in Backfill or recalculate data for multiple analyses.

Note:
Event frames are automatically deleted and recalculated for event frame generation analyses. Be aware that information such as annotations, acknowledgments, reason codes on those event frames will be lost.

4. Click **Start** to start the backfill or recalculation.
5. For details on the backfill or recalculation status or to cancel, right-click the analysis and click **Backfill/Recalculate Status**.

Video

For information on how to backfill or recalculate data for an analysis, watch this video:

https://www.youtube.com/watch?v=m7F0FwEykpw&rel=0

Backfill or recalculate data for multiple analyses

Before you start

PI Analysis Service 2016 R2 introduced recalculation capabilities for expression, rollup and SQC analyses. PI Data Archive storing the output PI points must be running version 2016 R2 or later. Beginning with PI Analysis Service 2018, you can opt for recalculation of all dependent analyses. An analysis whose input comes from an output of another analysis is considered dependent.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Optional. From **Analysis Searches**, select a search query and then select the result you want to operate upon. For example, select **Enabled**. Only the analyses that meet the criterion ( ) are displayed.
4. In the Analyses pane, select the analyses you want to backfill or recalculate.
5. In the Operations pane, click **Backfill/Recalculate # selected analyses**.

   Additional selection criteria are displayed.
6. In the **Start Time** and **End Time** fields, specify the time period for which you want to backfill missing data while retaining existing data, or delete and recalculate data. You can use standard PI time expressions, or click to select a specific date period.
7. Choose from the following actions:

To backfill or recalculate ...	Do this ...
Expression, rollup, or SQC analyses only	Decide how existing data should be treated: ◦ To ensure that existing data is retained and only missing data is backfilled, select **Leave existing data and fill in gaps**. ◦ If input data or analysis algorithms have changed and you want to delete existing output data, select **Permanently delete existing data and recalculate**.
A mix of analyses that includes event frame generation analyses or More than a page of analyses selections	Perform one of the following actions: ◦ Keep **Leave existing data and fill in gaps** selected and acknowledge that event frames in the time range will be permanently deleted along with all annotations on those event frames. ◦ Select **Permanently delete existing data and recalculate**. All annotations on event frames will be lost. ▪ Optional. Check the box next to **Recalculate dependent analyses** to queue all dependent analyses for manual recalculation. **Note:** ◦ Selecting this option may cause recalculation to occur multiple times for dependent analyses ◦ The recalculation time range for dependent analyses may be different than the time range you selected when queuing recalculation

Note:

Three factors affecting the recalculation time range of dependent analyses

- Time range of the original analysis
- How the output of the original analysis is used in the dependent analyses
- Whether or not the output time override is configured in the original analysis

8. Click **Queue** to start multiple backfills or recalculations.
9. Review the **Pending Operations** pane for details on the backfill or recalculation status.

Reconciliation of event frames during backfilling

When PI Analysis Service is restarted after short down-times, it initiates automatic backfilling concurrently with real-time analysis.

> **Note:**
> With PI Server 2015 and PI Server 2015 R2, PI Analysis Service does not initiate the automatic backfilling process if down-time is greater than 72 hours. With PI Server 2016, you can configure the **`MaximumAllowedAutoBackfillingSpanInHours`** configuration parameter in the PI Analysis Service to specify the desired time.

With PI Server 2015 R2, when PI Analysis Service restarts, it closes event frames that were open, indicating this reason in the event frame description field. When automatic backfilling for analyses is completed, Asset Analytics determines if there are event frames to be reconciled and, accordingly, starts the reconciliation phase. Event frame reconciliation may be necessary for any in-progress event frames that cannot be closed within the auto-backfilling time range because no closing event can be found. These event frames may be generated in either of these ways:

- The event frame may be already present when the analysis service was stopped
- The event frame may be newly created during auto-backfilling

Such in-progress event frames may have to be reconciled with those generated during real-time evaluation.

When you select an analysis using the Analysis tab or the Analysis Management plug-in, an orange icon ( ) in the analysis window indicates that the event frame reconciliation is pending completion. Hover over the icon to see details such as the number of events being processed, start and end times, and the time of last evaluation. When reconciliation is complete, the icon turns from orange to green.

Configure automatic recalculation of analyses and analyses templates

Before you start

From PI AF 2017 R2, you can automatically recalculate analyses. Although automatic recalculation is globally enabled by default, you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation. To configure automatic recalculation for a particular analysis (template), follow this procedure.

> **Note:**
> - PI AF Server and PI Analysis Service 2017 R2 or later are required.
> - Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Procedure

1. Go to the analysis (template) for which you want to enable automatic recalculation.
2. Click **Advanced** button at the bottom of the pane. Advanced options window opens.
3. Check the box next to **Recalculate analysis for out-of-order input events** and click **OK**.

   > **Note:**
   > You cannot opt in or out of automatic recalculation by accessing the analysis if it is based on a template. Go to the template to make changes.

Enable or disable automatic recalculation for multiple analyses

Before you start

For analyses writing to attributes (expression, rollup and SQC), automatic recalculation is available for PI Analysis Service 2017 R2 or later. In addition, PI AF Server 2017 R2 or later is required.

> **Note:**
> Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Optional. From **Analysis Searches**, select a search query and then select the result you want to operate upon. For example, select **Enabled**. Only the analyses that meet the criterion ( ) are displayed.
4. In the Analyses pane, select the analyses you want to enable or disable automatic recalculation.
5. In the Operations pane, click either **Enable automatic recalculation for selected analyses** or **Disable automatic recalculation for selected analyses**.
6. Check the box next to acknowledgement and click **Enable** or **Disable**.
7. Review the **Pending Operations** pane to track the progress of enabled or disabled operation.

Open recalculation log folder

To access a comprehensive log file of queued automatic/manual backfilling and recalculation requests for troubleshooting, follow the procedure.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Right-click anywhere in the **Operations** panel and click **Open Recalculation Log folder**. A folder containing a csv file with queued backfill/recalculation opens.

Management of analyses in a PI AF database

The Management feature (which includes analyses and notification rules) is available if your installation of PI System Explorer includes the Management plug-in.

For more information, go to Management of analyses. To search for analyses in your database, see Search for analyses or notification rules. For information on notifications management, see Management of notification rules.

Expression functions reference

Expression functions available for use in expression and event frame generation analyses are listed alphabetically. These functions follow performance equation (PE) syntax.

Topics in this section

- Abs
  Return the absolute value of an integer or real number.
- Acos
  Return the inverse cosine (arccos) of an integer or real number.
- AND
  Operator for the logical conjunction of two expressions.
- ArrayLength
  Get the length of an array.
- Ascii
  Return the ASCII character code of the first character of a string.
- Asin
  Return the inverse sine (arcsin) of a number.
- Atn
  Return the inverse tangent (arctan) of a number.
- Atn2
  Return the arctan for a specified tangent value a/b.
- Avg
  Return the average of one or more values.
- BadVal
  Test a value to see if it is bad.
- Bod
  Return a timestamp for beginning of the day from a time expression.
- Bom
  Return a timestamp for midnight on the first day of the month from a given time expression.
- Bonm
  Return a timestamp for midnight on the first day of a following month from a given time expression.
- Ceiling
  Return the smallest integer not less than x.
- Char
  Return a string built from specified ASCII character codes.
- Compare
  Compare two strings using wildcard characters ("*" and "?").
- Concat
  Concatenate two or more strings.
- Contains
  Determine if first string contains second string and return `True` or `False`.

- Convert
  Convert a value from its current unit of measure (UOM) to a specified UOM.
- Cos
  Return the trigonometric cosine of an integer or real number.
- Cosh
  Return the hyperbolic cosine of a number.
- Cot
  Return the trigonometric cotangent of a number
- Coth
  Return the hyperbolic cotangent of a number.
- Cov
  Determine the covariance between two sets of values given by attributes over a specified time range.
- Csc
  Return the trigonometric cosecant of a number.
- Csch
  Return the hyperbolic cosecant of a number.
- Curve
  For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$.
- Day
  Extract the day of the month from a time expression.
- DaySec
  Return the total time in seconds between the start of day (midnight) and the time denoted in the argument.
- DeltaValue
  Return the difference between the current and immediately previous values or evaluations of an attribute with a numeric or DateTime value type.
- DigState
  Translate a character string representing a digital state into its corresponding digital state.
- DigText
  Obtain the text corresponding to the digital state.
- E
  Constant function that returns the value for **e** (the base of the natural logarithm).
- ELSE
  Operator that returns the second of two specified values when the conditional expression in IF-THEN-ELSE statement is True.
- EventCount
  Find the number of good events for an attribute during a specified time range.
- EventFrame
  Return the value of event frame properties.
- Exp
  Return the exponential of an integer or real number.

- FilterData
  Get all the values in an array that satisfy a given condition.
- FindEq
  Find the first time within a time interval when an attribute is equal to a specified value.
- FindGE
  Find the first time within a time interval when an attribute is greater than or equal to a specified value.
- FindGT
  Find the first time within a time interval when an attribute is greater than a specified value.
- FindLE
  Find the first time within a time interval when an attribute is less than or equal to a specified value.
- FindLT
  Find the first time within a time interval when an attribute is less than a specified value.
- FindNE
  Find the first time within a time interval when an attribute is not equal to a specified value.
- FirstValue
  Get the first value in an array that satisfies a given condition.
- Float
  Convert a string to a number.
- Floor
  Return the largest integer not greater than *x*.
- Format
  Convert a number to string according to a format expression.
- Frac
  Return the fractional part of a real number. Returns 0 for integers.
- HasChanged
  Return `True` if an attribute has had any event in the specified time period; otherwise return `False`.
- HasValueChanged
  Determine if the value of an attribute or expression has changed since last evaluated during an analysis.
- Hour
  Extract the hour from a time expression.
- IF
  The condition for an IF-THEN-ELSE statement.
- IN
  Operator that indicates whether a value is included in a set or range of values.
- InStr
  Determine the location within a string where a sub-string match is first found.
- Int
  Return the integer part of an integer or real number.

- InterpolatedValues
  Obtain interpolated values over a specified time range using the specified time step.
- IsSet
  For an attribute associated with a PI point, determine if the point is annotated, substituted, or questionable.
- LastValue
  Get the last value in an array that satisfies a given condition.
- LCase
  Convert a string to a lowercase string.
- Left
  Determine a specified number of characters of a string from the left.
- Len
  Determine the length of a string.
- LinRegr
  Return a one-based array of slope, intercept and R^2 based on time stamped values.
- Log
  Return the natural logarithm (base e = 2.7182818...) of an integer or real number.
- Log10
  Return the base 10 logarithm of an integer or real number.
- Logbase
  The logarithm of a positive numeric value to a specified base.
- LTrim
  Remove the leading blanks from a string.
- MapData
  Apply a function to each value in an array.
- Max
  Return the maximum of a set of values.
- Median
  Return the median (middle) value for a set of one or more values.
- Mid
  Return a sub-string within a string.
- Min
  Return the minimum of a set of values.
- Minute
  Extract the minute from a time expression.
- Mod
  Operator that returns the remainder from the quotient of two numeric values.
- Month
  Extract the month from a time expression.
- NextEvent
  Find the timestamp of the next recorded value after a specified time.
- NextVal
  Find the next recorded value after a specified time.

- Noon
  Return a timestamp for noon on the day of a given time expression.
- NoOutput
  Do not write current calculation result.
- Normalrnd
  Return a random number that maps the normal distribution curve using a specified mean and standard deviation.
- NOT
  Operator that returns the negation of an expression.
- NumOfChanges
  Return the number of changes in value for an attribute within a specified time range.
- OR
  Operator for the logical disjunction of two expressions.
- ParseTime
  Translate a PI time expression to a timestamp.
- PctGood
  Find the time percentage, over a specified time range, for which the attribute had good values.
- Pi
  Constant function that returns the value for pi.
- Poisson
  Return a random number that maps a Poisson distribution with a specific mean.
- Poly
  Return the polynomial $c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$.
- Pow
  Return **x** raised to the power **y**.
- PrevEvent
  Find the timestamp of the last recorded value before a specified time.
- PrevVal
  Find the last recorded value before a specified time.
- PStDev
  Return the *population standard deviation* for a population of one or more values.
- Rand
  Returns a random number from a specified range.
- Range
  Find the difference between the maximum and minimum values for an attribute during a specified time range.
- Rate
  Return the difference over time between the current value and a previously evaluated value of an attribute or (time) expression.
- RecordedValues
  Obtain values within a particular time range based on user-specified boundary type.

- RecordedValuesByCount
  Obtain a specified number of values beginning at the requested start time in the direction specified.
- Remainder
  Return the remainder resulting from the division of x by y using the IEEERemainder method.
- Right
  Determine a specified number of characters of a string from the right.
- Round
  Round a number or time to the nearest unit.
- Roundfrac
  Round a value to a specified number of decimal places.
- RTrim
  Trim trailing blanks from a string.
- Sec
  Return the trigonometric secant of a number.
- Sech
  Return the hyperbolic secant of a number.
- Second
  Extract the second from a time expression.
- SecSinceChange
  Return the time in seconds since an attribute value was updated.
- Sgn
  Return an indicator of the numerical sign of a number.
- Sin
  Return the trigonometric sine of a number.
- Sinh
  Return the hyperbolic sine of a number.
- Sqr
  Return the square root of a number.
- StateNo
  Translate a digital state into its corresponding enumeration value.
- SStDev
  Return the *sample standard deviation* for two or more arguments that are a sample of a larger population.
- StDev
  Find the time-weighted standard deviation of values for an attribute from a specified time range.
- String
  Convert any value to a string.
- TagAvg
  Find the time-weighted average of values for an attribute during a specified time range.

- TagBad
  Test the point for an abnormal state at a specified time
- TagDesc
  For an attribute associated with a PI point, retrieve that point's descriptor from the point database.
- TagEU
  For an attribute associated with a PI point, retrieve the point's engineering unit string from the point database.
- TagExDesc
  For an attribute associated with a PI point, retrieve the point's extended descriptor from the point database.
- TagMax
  Find the maximum value of an attribute during a specified time range.
- TagMean
  Find the average (mean) of values for an attribute during a specified time range.
- TagMin
  Find the minimum value of an attribute during a specified time range.
- TagName
  For an attribute associated with a PI point, get a point's name from the point database.
- TagNum
  For an attribute associated with a PI point, get a point's number from the point database.
- TagSource
  For an attribute associated with a PI point, get a point's point source string from the point database.
- TagSpan
  For an attribute associated with a PI point, get a point's span from the point database.
- TagTot
  Find the totalized value (time integral) of an attribute's values over a specified time range.
- TagType
  For an attribute associated with a PI point, get a point's point type from the point database.
- TagTypVal
  For an attribute associated with a PI point, get a point's typical value from the point database.
- TagVal
  Return the value of an attribute at a specified time.
- TagZero
  For an attribute with a PI point data reference, get a PI point's "zero" value from the point database.
- Tan
  Return the trigonometric tangent of a number.
- Tanh
  Return the hyperbolic tangent of a number.
- Text
  Concatenate strings representing argument values.

- THEN
  Operator that returns the first of two specified values when the conditional expression in IF-THEN-ELSE statement is True.
- TimeEq
  Find the total time, within a range, when an attribute value is equal to a specified value.
- TimeGE
  Find the total time, within a range, when an attribute value is greater than or equal to a specified value.
- TimeGT
  Find the total time, within a range, when an attribute value is greater than a specified value.
- TimeLE
  Find the total time, within a range, when an attribute value is less than or equal to a given value.
- TimeLT
  Find the total time, within a range, when an attribute value is less than a specified value.
- TimeNE
  Find the total time, within a range, when an attribute value is not equal to a specified value.
- TimeStamp
  Return the time stamp for a single time-stamped value or an array of time-stamped values.
- Total
  Return the sum of two or more values.
- Trim
  Trim blanks on both sides of a string.
- Trunc
  Truncate a number or time to the next lower unit.
- UCase
  Convert a string to an uppercase string.
- Weekday
  Extract the day of the week from a timestamp.
- Year
  Extract the year from a time expression.
- Yearday
  Extract the day of the year from a time expression.

Abs

Return the absolute value of an integer or real number.

Syntax

```
Abs(x)
```

Arguments

- **x**

  An integer or real number

Returns

The absolute value of *x*

Exceptions

Return an error value if *x* is not an integer or real number

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Abs(-2.2)`

  [Returns 2.2]

- `Abs('att1')`

  [Return the absolute value of 'att1' at trigger time]

Acos

Return the inverse cosine (arccos) of an integer or real number. The inverse cosine of *x* is the angle in radians whose cosine is equal to *x*.

Syntax

```
Acos(x)
```

Arguments

- **x**

  Must be a real number between -1.0 and 1.0, inclusive

Returns

The inverse cosine of *x*, in radians

Exceptions

If *x* is not a number, or is less than -1.0 or greater than 1.0, returns an error value

Example

- `Acos(-0.5)`

  [Returns 2.0944]

- `Acos(0.75)`

  [Returns 0.72273]

AND

The logical conjunction of two expressions that returns **`True`** if both expressions are true and **`False`** otherwise.

Syntax

```
expression1 AND expression2
```

Arguments

- **`expression1, expression2`**

  Any expression that evaluates to true or false

Returns

`True` if both expressions are true (non-zero) and **`False`** (zero) otherwise

Exceptions

None

Example

- `('att1' > 50) AND ('att2' = "good")`

ArrayLength

Get the length of an array.

Syntax

```
ArrayLength(array)
```

Arguments

- **`array`**

  a variable representing the array to operate on

Returns

The number of values in the array

Exceptions

None

Notes

None

Example

- `ArrayLength(Data)`

  [Return the number of values in the array named **`Data`**]

Ascii

Return the ASCII character code of the first character of a string.

Syntax

```
Ascii(s1)
```

Arguments

- **s1**

  Any expression evaluating to a string

Returns

The character code of the first character of the string

Exceptions

If the argument is not a string, an error value is returned

Example

- `Ascii("D")`

  [Returns 68, the ASCII character code for D]

- `Ascii("Program")`

  [Returns 80, the ASCII character code for the first letter of the string]

Asin

Return the inverse sine (arcsin) of a number. The inverse sine of *x* is the angle in radians whose sine is equal to *x*.

Syntax

```
Asin(x)
```

Arguments

- **x**

  Must be a real number between -1.0 and 1.0, inclusive

Returns

The inverse sine of *x*, in radians

Exceptions

If *x* is not a number, or is less than -1.0 or greater than 1.0, an error value is returned

Example

- `Asin(-0.5)`

  [Returns -0.5236]

- `Asin(TagVal('att1','y+8h'))`

  [Return the inverse sine of the value of 'att1' at 8am yesterday]

- `Asin('att1')`

[Return the inverse sine of the value of 'att1' at trigger time]

Atn

Return the inverse (or arc) tangent of an integer or real number. The inverse tangent of *x* is the angle in radians whose tangent is equal to *x*.

Syntax

```
Atn(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The inverse tangent of *x*, in radians

Exceptions

Returns an error if *x* is not an integer or real number

Example

- `Atn(1)`

  [Returns 0.7854]

- `Atn(-2.2)`

  [Returns -1.1442]

- `Atn('att1')`

  [Return the inverse tangent of the value of 'att1' at trigger time]

- `Atn('att1' - 'att2')`

  [Return the inverse tangent of the difference of 'att1' and 'att2' at trigger time]

Atn2

Return the inverse tangent (arctan) of a tangent value a/b. The inverse tangent is the angle measured in radians from the positive x-axis to a line whose endpoints are the origin and the Cartesian coordinates (b, a).

Syntax

```
Atn2(x, y)
```

Arguments

- **x**

  An integer or real number

- **y**

  A non-zero integer or real number

Returns

The inverse tangent in radians of the tangent value x/y

Exceptions

Returns an error if *x* or *y* is not an integer or real number

Example

- `Atn2(1,2)`

  [Returns 0.46365]

- `Atn2('att1', 'att2')`

  [Return the inverse tangent in radians of the tangent value 'att1'/'att2' at trigger time]

- `Atn2(TagVal('att1','y+8h'), TagVal('att2', 'y+8h'))`

  [Return the inverse tangent in radians of the tangent value 'att1'/'att2' at 8am yesterday]

Avg

Return the average from a set of one or more values.

Syntax

```
Avg (x1 [, ... xn])
```

Arguments

- **x1, ... xn**

  Arguments or a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The average of one or more values. The result is of the same data type as arguments.

Exceptions

Arguments whose run-time values are character strings or digital states are not included in the average. If all values are character strings or digital states, Avg returns an error value. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs. Avg returns not time-based but event-based average.

Example

- `Avg(14, 'att1', 14.5, TagVal('att2', '14-Dec-16'))`

[Find the average of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at the start of day (12:00am) on Dec 14, 2016]

- `Avg('*', 'y+8h', 'Saturday')`

  [Find the average of these time stamps: now, 8:00am yesterday, and start of day (12:00am) last Saturday]

- `Avg('*'-'*-1h', 't+4h'-'y+8h', 'y+2h'-'20')`

  [Find the average of these time periods: from 1 hour ago to now, from 8:00am yesterday to 4:00am today, and 2:00am yesterday to the start of day (12:00am) on the 20th of this month]

- `Avg(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Avg(Variable1)	5

  [Find the average of values in an array `Variable1`]

BadVal

Test a value to see if it is bad. For an attribute associated with a PI point, any system digital state value is considered bad (`No Data` or `Calc Failed`, for example).

Syntax

```
Badval(x)
```

Arguments

- **x**

  any expression evaluating to a value

Returns

`True` if the value is bad

`False` if the value is not bad

Exceptions

Returns `True` for blob PI points. Returns `False` for string PI points

Example

- `BadVal(1/0)`

  [Returns `True`]

- `BadVal(10)`

  [Returns `False`]

- `BadVal('att1')`

  [Returns `True` if the value of 'att1' is bad]

- `BadVal(FindEq('att1', 't', '*', 10))`

  [Returns `True` if the embedded function (**`FindEq`** in this example) has no result or has error for any reason]

Bod

Return a timestamp for beginning of the day from a time expression.

Syntax

```
Bod(t1)
```

Arguments

- **`t1`**

  A time expression, enclosed in single quotes

Returns

Timestamp for the start of the day

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bod('*')`

  [Return a timestamp for beginning of the day today.]

- `Bod('y')`

  [Return a timestamp for beginning of the day yesterday.]

- `Bod(FindEq('att1', '-3d', '*', 50))`

  [Return a timestamp for beginning of the day when 'att1' value was first equal to 50 in the past 72 hours.]

Bom

Return a timestamp for midnight on the first day of the month from a given time expression.

Syntax

```
Bom(t1)
```

Arguments

- **t1**

  A time expression

Returns

Timestamp for the start of the month.

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bom('*')`

  [Return a timestamp for midnight on the first day of this month.]

- `Bom(PrevEvent('att1', '*'))`

  [Return a timestamp for midnight on the first day of the month when 'att1' had a value before the current one.]

- `Bom(FindEq('att1', '-60d', '*', 50))`

  [Return a timestamp for midnight on the first day of the month when the value of 'att1' was first equal to 50 in the past 60 days.]

Bonm

Return a timestamp for midnight on the first day of a following month from a given time expression.

Syntax

```
Bonm(t1)
```

Arguments

- **t1**

  Time expression, enclosed in single quotes

Returns

Timestamp for the start of the next month.

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bonm('*')`

  [Return a timestamp for midnight on the first day of next month.]

- `Bonm('y')`

  [Return a timestamp for midnight on the first day of the following month from yesterday's date.]

- `Bonm(FindEq('att1', '-60d', '*', 50))`

  [Return a timestamp for midnight on the first day of the following month when the value of 'att1' was first equal to 50 in the past 60 days.]

Ceiling

Return the nearest integer greater than or equal to *x*.

Syntax

```
Ceiling(x)
```

Arguments

- **x**

  integer or a real number, an attribute whose value evaluates to a number

Returns

The nearest integer greater than or equal to *x*

Exceptions

Returns an error if *x* is a digital state, time expression, time interval or a string

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Ceiling(2.3)`

  [Returns 3]

- `Ceiling(-2.3)`

  [Returns -2]

- `Ceiling(TagVal('att1', '12/30/16'))`

  [Return the nearest integer to the value of 'att1' at the start of day (12:00am) on December 30, 2016]

Char

Build a string from ASCII character codes.

Syntax

```
Char(x1, ... xn)
```

Arguments

- **`x1, ... xn`**

  Integers

Returns

A string built from the 80 character codes

Exceptions

Returns an error if an argument is not a number

Example

- `Char(80, 73)`

  [Returns "PI"]

- `Char(65)`

  [Returns "A"]

- `Char(5 * 10)`

  [Returns "2"]

Compare

Compare two strings using wildcard characters ("*" and "?") or compare two operands using logical operators and deadband value.

Syntax

```
Compare {[(s1, s2 [, caseSensitive])]|[x1, x2, op, deadband]}
```

Arguments

- **`s1, s2`**

  string (s2 can contain wildcard characters)

- **`caseSensitive`**

  Optional flag indicating if the comparison is case sensitive. If `False` (the default), the comparison is not case-sensitive. If `True`, the comparison is case-sensitive.

- **`x1, x2`**

  Numeric value (integer or real number) or time expression

- **op**

  An operator used for comparison. Must be one of the following operators, or a variable defined as one of these: <, >, <=, >=

- **deadband**

  Numeric value (integer or real number) or timespan. *Deadband* specifies a buffer (tolerance) around *x2* and prevents repeated alerts when the value fluctuates within the *deadband* range.

Returns

- `True` if *s1* = *s2*
- `True` if *x1* <op> *x2* is true, and uses *deadband* value for subsequent evaluations

`False` otherwise

Exceptions

Wildcard characters in *s1* are treated literally and not as wildcards.

The deadband value is applied to *x2*, not *x1*.

Example

- `Compare('att1', 100, "<=", 5)`

  [Returns `True` if 'att1' is less than or equal to 100, and uses the deadband value of 5 for subsequent evaluations]

- `Compare("What", "what", True)`

  [Returns `False`]

- `Compare("b", "a")`

  [Returns `False`]

- `Compare("What", "wha?")`

  [Returns `True`]

- `Compare("What", "wh")`

  [Returns `False`]

Concat

Concatenate two or more strings.

Syntax

```
Concat(s1, s2 [, ... sn])
```

Arguments

- **s1, ... sn**

  Must be character strings, or expressions yielding character strings.

Returns

The character strings, concatenated together. This function does not insert blanks between its arguments. To include a space in the concatenated string, add an argument consisting of a string that has a single space enclosed in double quotes.

Example

- `Concat("shut", "down")`

  [Returns "shutdown"]

- `Concat("shut ", "down")`

  [Returns "shut down"]

Contains

Determine if first string contains second string and return `True` or `False`.

Syntax

```
Contains(s1, s2 [,caseSensitive])
```

Arguments

- **`s1`**

  Primary string

- **`s2`**

  Substring searched for in primary string

- **`caseSensitive`**

  Optional: Boolean that indicates whether the search is case sensitive. If `False` (or omitted), the search is not case sensitive. If `True`, the search is case sensitive

Returns

`True` if the primary string (*s1*) contains the substring (*s2*) and `False` otherwise

Exceptions

If either of the first two arguments (*s1* or *s2*) is not a string, the function returns an error value

If the optional third argument (*caseSensitive*) is not a Boolean, the function returns an error value

Example

- `Contains("What","Hat",True)`

  [Returns `False`]

- `Contains("What","Hat")`

  [Returns `True`]

- `Contains("What","at")`

[Returns `True`]

- `Contains("hat","what")`

  [Returns `False`]

Convert

Convert a value from its current unit of measure (UOM) to a specified UOM; for a value with no UOM, assign the specified UOM.

Syntax

```
Convert(x, toUnit)
```

Arguments

- **`x`**

  Any expression that resolves to a numeric value, including the name of a numeric attribute enclosed in single quotation marks, a variable name, or a constant value

- **`toUnit`**

  The output UOM enclosed in double quotation marks

Returns

The numeric value converted to the specified UOM

Exceptions

Returns an error if *toUnit* and the initial UOM for *x* are not in the same UOM class

Example

- `Convert('MetricWeight', "lb")`
- `Convert('Ambient Temperature', "degC")`

  Note:
  PI AF substitutes `deg` with `°`, if not otherwise found in the lookup.

Cos

Return the trigonometric cosine of an integer or real number.

Syntax

```
Cos(x)
```

Arguments

- **`x`**

  Must be an integer or real number, which represents an angle in radians

Returns

The cosine of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Cos(1.1)`

  [Returns 0.4536]

- `Cos(1)`

  [Returns 0.5403]

- `Cos('att1')`

  [Return the trigonometric cosine of the value of 'att1' at trigger time]

Cosh

Return the hyperbolic cosine of a number.

Syntax

```
Cosh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cosine of *x*

Exceptions

If *x* is not an integer or real number, returns an error

Example

- `Cosh(1)`

  [Returns 1.5431]

- `Cosh(-1)`

  [Returns 1.5431]

- `Cosh('att1')`

  [Return the hyperbolic cosine of the value of 'att1' at trigger time]

Cot

Return the trigonometric cotangent of a number.

Syntax

```
Cot(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The cotangent of *x*

Exceptions

If *x* is not a number, returns an error

Example

- `Cot(1)`

  [Returns 0.64209]

- `Cot(1.1)`

  [Returns 0.50897]

- `Cot('att1')`

  [Return the trigonometric cotangent of the value of 'att1' at trigger time]

Coth

Return the hyperbolic cotangent of a number.

Syntax

```
Coth(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cotangent of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Coth(1)`

[Returns 1.313]

- `Coth(1.1)`

  [Returns 1.2492]

- `Coth('att1')`

  [Return the hyperbolic cotangent of the value of 'att1' at trigger time]

Cov

Determine the covariance between two sets of values given by attributes over a specified time range.

Syntax

```
Cov(x, y, mode, starttime, endtime, type [, pctgood])
```

Arguments

- **`x`**

  an attribute with the first set of time series data (such as PI point data reference) enclosed in single quotes

- **`y`**

  an attribute with the second set of time series data (such as PI point data reference) enclosed in single quotes

- **`mode`**

  a number that specifies how to align time-stamped values

  choose from 0, 1 or 2 where:

  0 represents the combination of time-stamped values from *x* and *y*

  1 represents values from both attributes according to *x*'s time stamps

  2 represents values from both attributes according to *y*'s time stamps

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute *endtime*

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute *starttime*

- **`type`**

  a number specifying the covariance type: use 0 for "population" and 1 for "sample"

- **`pctgood`**

  Optional. Minimum percentage of time during the time range that attribute values must be good.

You can set *pctgood* as a threshold to ensure that there are sufficient good values to calculate **Cov**.

Returns

Covariance of *x* and *y*

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value

Notes

Bad values are excluded from **Cov** calculation

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good

Example

- `Cov('att1', 'att2', 0, 't', '+1h', 1, 80)`

  [Return the sample covariance of 'att1' and 'att2' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `Cov('att1', 'att2', 1, 't', '+1h', 1)`

  [Return the population covariance of 'att1' and 'att2' based on time stamps of 'att1' between 12:00 and 1:00am today]

Csc

Return the trigonometric cosecant of a number.

Syntax

```
Csc(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The cosecant of *x*

Exceptions

If *x* is not a number, returns an error

Example

- `Csc(1)`

[Returns 1.1884]

- `Csc(1.1)`

  [Returns 0.7487]

- `Csc('att1')`

  [Return the trigonometric cosecant of the value of 'att1' at trigger time]

Csch

Return the hyperbolic cosecant of a number.

Syntax

```
Csch(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cosecant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Csch(1)`

  [Returns 0.85092]

- `Csch(1.1)`

  [Returns 0.748]

- `Csch('att1')`

  [Return the hyperbolic cosecant of the value of 'att1' at trigger time]

Curve

For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$.

Syntax

```
Curve(x, (x1,y1) (x2,y2) … (xn,yn) )
```

Arguments

- **x**

  Expression evaluating to a number

- **x_1, y_1**

  The first point on the curve. The x_i's and y_i's are numeric constants evaluated at compile time. The values set for x_i's must be in ascending order.

Returns

For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$. If the value of *x* is less than x_1 then y_1 is returned and if it is greater than x_n, y_n is returned. The points are assumed to be ordered in the *x* direction from smallest to largest.

Exceptions

If the value of *x* is not an integer or real number, an error value is returned

Example

- `Curve(3, (0,100) (100,0) )`

  [Returns 97]

- `Curve('att1', (25,25) (75,75) )`

Day

Extract the day of the month from a time expression.

Syntax

```
Day(t1)
```

Arguments

- **t1**

  time expression enclosed in single quotes

Returns

The day of the month of a time expression in the range 1 through 31.

Exceptions

None

Example

- `Day('*')`

  [Return what day of the month today is.]

- `Day('y')`

  [Return what day of the month yesterday was.]

- `Day(FindEq('att1', '-28d', '*', 50))`

  [Return what day of the month it was when the value of 'att1' was first equal to 50 in the past 28 days .]

DaySec

Return the total time in seconds between the start of day (midnight) and the time denoted in the argument.

Syntax

```
DaySec(t1)
```

Arguments

- **t_1**

  A time expression, enclosed in single quotes

Returns

Total seconds since the start of day (midnight) till *t1*, in the range 0-86399

Exceptions

None

Example

- `DaySec('*')`

  [Return the number of seconds from the start of day (midnight) until now]

DeltaValue

Return the difference between the current and immediately previous values or *evaluations* of an attribute with a numeric or DateTime value type.

Syntax

```
DeltaValue(x [, prevEval])
```

Arguments

- **x**

  An attribute or expression of a numeric or DateTime value type

- **prevEval**

  (Boolean) If `False` or not specified, *x* must be an attribute, and **`DeltaValue`** returns the difference between its current value and immediately previous value.

  If `True`, **`DeltaValue`**returns the difference between the current and immediately previous *evaluations* of *x*, which can be any expression with a numeric or DateTime value type

Returns

Returns the difference between the current value and the immediately previous value of an attribute with a numeric or DateTime value type, or the difference between the current and immediately previous *evaluations* for any expression with a numeric or DateTime value type

Notes

When *x* is of DateTime data type, **`DeltaValue`** returns the time interval (in seconds) between two DateTime values. If the time interval value is output to an attribute, make sure the attribute data type is set to double

This function may use input values from PI points. If the compression is on for a PI point, all of its values may not be preserved in PI Data Archive. Before you try to validate the results of this function using a client tool such as PI Datalink, make sure the compression is turned off. This is to ensure that all associated PI point values are included for validation. When validation is complete, be sure to turn the compression back on again as a best practice

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `DeltaValue('att1' [, FALSE])`

  [Return the difference between the current and immediately preceding value of 'att1']

- `DeltaValue('att1', TRUE)`

  [Return the difference between the current and last *evaluated* value of 'att1']

DigState

Convert a string into a corresponding PI Data Archive digital state object, either based on the attribute's digital state enumeration or on a system enumeration set values (for example, `Calc Failed`). Use **`DigState`** embedded in other functions when PI Data Archive digital state object is required for input (such as with **`StateNo`**) or in conjunction with other functions for comparison with digital state attributes.

Syntax

```
DigState(s1 [, x])
```

Arguments

- **s_1**

  A string representing a digital state in double quotes

- **x**

  Optional: An attribute with PI point digital state reference or of value type enumeration set. If omitted, only the system digital set is searched for the given string.

Returns

An enumeration value

Exceptions

If the string does not represent a digital state of the specified attribute, the function returns `Calc Failed`. If the attribute is omitted and string does not represent a system digital state, `Calc Failed` is returned.

Example

- `'att1' > DigState("Program", 'att1')`

  [Returns `True` if current digital state of 'att1' is greater than the digital state represented by "Program" value]

- `DigState("No Result")`

  [Construct a value with system digital state `No Result`]

DigText

Obtain the text corresponding to the digital state.

Syntax

```
DigText(digstate)
```

Arguments

- **`digstate`**

  An argument that evaluates to a digital state

Returns

The text for the digital state

Exceptions

Returns an error if the argument is not a digital state

Example

- `DigText(TagVal('enum_att1', 'y'))`

  [Returns the digital state value in string]

- `DigText('enum_att1')`

  [Returns digital state value in string. Returns an error message if 'enum_att1' is not a PI point attribute with digital state value]

E

Return the value for **e** (the base of the natural logarithm).

Syntax

```
E()
```

Arguments

None

Returns

The value for **e** (the base of the natural logarithm).

Exceptions

None

Example

- `E()`

ELSE

Operator that returns the second of two specified values when the conditional expression in IF-THEN-ELSE statement is True.

Syntax

```
IF (expression) THEN (x) ELSE (y)
```

Arguments

- **`expression`**

  Any expression that evaluates to true or false

- **`x, y`**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return the value of 'att2'; otherwise return the value of 'att3']

EventCount

Find the number of good events for an attribute during a specified time range.

Syntax

```
EventCount(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of good values for events within the specified time interval.

Returns

Number of events for the attribute during a specified time range.

Exceptions

If there are no good values or the **pctgood** minimum is not reached for the given time interval, an error is returned.

Notes

Bad values are excluded from **EventCount** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `EventCount('att1', 't', '+1h')`

  [Return the count of events between 12:00 and 1:00am today.]

- `EventCount('att1', 't', '+1h', 80)`

  [Return the count of events between 12:00 and 1:00am today when at least 80% of the values were good.]

EventFrame

Return the value of event frame properties.

Syntax

```
EventFrame(parameter)
```

Arguments

- **parameter**

  parameter string *StartTime*, *EndTime* or *Duration* enclosed in double quotes; not case-sensitive

Returns

The value of the parameter

Exceptions

If the event frame is not active or still open, returns `No Results`

Notes

- This function is only to be used when configuring output expressions in an event frame generation analysis
- The output can be mapped to an attribute with value type DateTime for *StartTime* and *EndTime*. For *Duration*, select double (time returned in seconds)
- You can preview results by right-clicking the analysis

Example

- `EventFrame("Duration")`

  [Return the duration of an event frame. Time is returned in seconds]
- `TagAvg('att1', EventFrame("StartTime"), EventFrame("EndTime"))`

  [Find the time-weighted average of the values of 'att1' during an event frame]

Exp

Return the exponential of an integer or real number. This is the number e^x, where e = 2.7182818...

Syntax

```
Exp(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The exponential of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Exp(11)`

  [Returns 59874]
- `Exp('att1')`

  [Return the exponential of the value of 'att1' at trigger time]
- `Exp(TagVal('att1','t'))`

  [Return the exponential of the value of 'att1' at the start of day (12am) today]

FilterData

Get all the values in an array that satisfy a given condition.

Syntax

```
FilterData(array, condition($val))
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

An array of all the values that satisfy the given condition

Exceptions

None

Notes

None

Example

- `FilterData(Data, (NOT BadVal($val) AND $val > 50))`

  [From an array named **`Data`**, return a new array of values where none are bad values and all are greater than 50]

FindEq

Find the first time within a time interval when an attribute is equal to a specified value.

Syntax

```
FindEq(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  name of an attribute enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string)

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was equal to the specified value

Exceptions

If the attribute was never equal to the specified value, an error value is returned

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was equal to the specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindEq('att1', '-1h', '*', 40)`

  [Return the timestamp of the first time since an hour ago when ' att1' was equal to 40]

- `FindEq('enum_att1', '30-March-17', '*', "On")`

  [Return the timestamp of the first time since the start of day (12am) on March 30th, 2017 when enumeration value 'enum_att1' was equal to "On"]

FindGE

Find the first time within a time interval when an attribute is greater than or equal to a specified value.

Syntax

```
FindGE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was greater than or equal to the specified value.

Exceptions

If the attribute was always less than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was greater than or equal to the specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindGE('att1', 'y', '+2h', 40)`

  [Return the timestamp between midnight and 2:00 am yesterday when 'att1' was first greater than or equal to 40.]

FindGT

Find the first time within a time interval when an attribute is greater than a specified value.

Syntax

```
FindGT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was greater than the specified value.

Exceptions

If the attribute was never greater than the specified value, an error is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was greater than he specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindGT('att1', 't', '*', 40)`

  [Return the timestamp of the first time after midnight today when 'att1' was greater than 40]

- `FindGT('att1', '-1h', '*', TagVal('att1', 'y+18h'))`

  [Return the timestamp of the first time since an hour ago when the value of 'att1' was greater than its value at 6pm yesterday]

FindLE

Find the first time within a time interval when an attribute is less than or equal to a specified value.

Syntax

```
FindLE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to *starttime* within the specified interval when the attribute was less than or equal to the specified value.

Exceptions

If the attribute was always greater than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was equal to the specified value.
- If *endtime* is earlier than *starttime*, the time interval is searched backwards.

Example

- `FindLE('att1', 't', '*', 40)`

  [Return the timestamp of the first time after midnight today when *att1* was less than or equal to 40.]

FindLT

Find the first time within a time interval when an attribute is less than a specified value.

Syntax

```
FindLT(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **`starttime`**

- **`x`**

  Must be an integer or real number or digital state (character string), the value to search for

Returns

The timestamp closest to **`starttime`** within the specified interval when the attribute was less than the specified value.

Exceptions

If the attribute was less than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was lower than the specified value.
- If **`endtime`** is earlier than **`starttime`**, the time interval is searched backwards.

Example

- `FindLT('att1', 'y', 't', 40)`

  [Return the timestamp of the first time between midnight yesterday and midnight today that 'att1' was less than 40.]

- `FindLT('att1', '-1h', '*', TagVal('att1', 'y+18h'))`

  [Return the timestamp of the first time since an hour ago when the value of 'att1' was less than its value at 6pm yesterday]

FindNE

Find the first time within a time interval when an attribute is not equal to a specified value. "No Data" events will be skipped.

Syntax

```
FindNE(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute enclosed in single quotation marks.

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **`starttime`**

- **`x`**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **`starttime`**, within the specified interval, for which the attribute was not equal to the specified value. "No Data" events will be skipped.

Exceptions

If the attribute was always equal to the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was not equal to the specified value.
- If **`endtime`** is earlier than **`starttime`**, the time interval is searched backwards.

Example

- `FindNE('att1', '-1h', '*', 40)`

[Return the timestamp of the first time since an hour ago when ' att1' was not equal to 40]

- `FindNE('enum_att1', '30-March-17', '*', "On")`

  [Return the timestamp of the first time since the start of day (12am) on March 30th, 2017 when enumeration value 'enum_att1' was not equal to "On"]

FirstValue

Get the first value in an array that satisfies a given condition.

Syntax

```
FirstValue(array [, condition($val)])
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

The first value that satisfies the given condition

Exceptions

None

Notes

None

Example

- `FirstValue(Data)`

  [Return the first value from an array named **`Data`**]

- `FirstVal(Data, $val < 90)`

  [From the array named **`Data`**, return the first value in the array that is less than 90]

Float

Convert a string to a number.

Syntax

```
Float(x)
```

Arguments

- **x**

  A string or a number, or an attribute whose value evaluates to a number at time of evaluation

Returns

A number for a numeric string. If *x* is already a number, *x* is returned

Exceptions

If *x* is not a number or a numeric string, returns `Calc Failed`

Notes

Unit of measure of the argument, if it exists, is carried over to the result. **`Float`** also takes *timespan* and *boolean* as argument. Note, however, that **`Float`** only converts *timespan* format to the number of seconds from 12:00am Jan 1, 1970.

Example

- `Float("12.3")`

  [Converts string to a number and returns 12.3]

- `Float(12.3)`

  [Returns 12.3]

- `Float('*')`

  [Return the total number of seconds passed since Jan 1, 1970]

Floor

Return the nearest integer less than or equal to *x*.

Syntax

```
Floor(x)
```

Arguments

- **x**

  Integer or a real number, or an attribute whose value evaluates to a number at time of evaluation

Returns

The nearest integer less than or equal to *x*

Exceptions

Returns an error if *x* is a digital state, time expression, time interval or a string

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Floor(3.14)`

  [Returns 3]

- `Floor(-3.14)`

  [Returns -4]

- `Floor(TagVal('att1', '*'))`

  [Return the nearest integer less than or equal to the current value of 'att1']

Format

Convert a number to string according to a format expression.

Syntax

```
Format(x, format [,culture])
```

Arguments

- **x**

  An integer or a real number

- **format**

  Format-control string, similar to that used by the C language function **Sprintf**

- **culture**

  Optional. Culture name string, enclosed in double quotation marks. Example values include "de-DE", "ja-jp", "pt-br", "zh-cn". Use "" to specify the culture-invariant setting (the default setting).

Returns

A formatted string

Example

- `Format(1234.567, "%10.2f", "de-DE")`

  [Returns "1234,57", a formatted string]

- `Format(45, "%3.3d")`

  [Returns "045"]

Frac

Return the fractional part of a real number. Returns 0 for integers.

Syntax

```
Frac(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The fractional part of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Notes

By definition: `Int(x) + Frac(x) = x`

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Frac(1)`

  [Returns 0]

- `Frac(1.3)`

  [Returns 0.3]

- `Frac(TagVal('att1', '*'))`

  [Return the fractional part of the value of 'att1' at current time.]

HasChanged

Return `True` if an attribute has had any event in the specified time period; otherwise return `False`.

Syntax

```
HasChanged(attname, t1)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks

- **t1**

  The start of a time period ending at execution time; can be any relative time expression in single quotation marks (such as 't - 4h')

Returns

`True` if *attname* has any event in the specified time period; otherwise returns `False`

Example

- `HasChanged('att1', 't-4h')`

[Returns `True` if 'att1' received any updates since 8 p.m. last night, regardless of whether the update changed the actual attribute value]

HasValueChanged

Determine if the value of an attribute or expression has changed since last evaluated during an analysis.

Syntax

```
HasValueChanged(x)
```

Arguments

- **x**

  attribute or expression

Returns

`True` if the value of *x* has changed since last evaluated during the analysis; otherwise returns `False`.

If *x* is an expression that this function has not yet evaluated during the analysis, then the function returns `False`.

If *x* is an attribute that this function has not yet evaluated during the analysis, then the function finds the previous recorded value and compares the current value to that previous value. If there is no previous value, then the function returns `False`.

Notes

If PI Analysis Service restarts during an analysis, any previous evaluations are lost.

Example

- `HasValueChanged('att1')`

  [Returns `True` if the value of *att1* has changed since last evaluated during the analysis]

- `HasValueChanged('att1'+'att2')`

  [Returns `True` if the value of sum of *att1* and *att2* has changed since last evaluated during the analysis]

Hour

Extract the hour from a time expression.

Syntax

```
Hour(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes.

Returns

The hour of time, in the range 0-23.

Exceptions

None

Example

- `Hour('*')`

  [Return the hour portion of current time.]

- `Hour('Saturday')`

  [Returns 0.]

- `Hour(FindEq('att1', '-1d', '*', 50))`

  [Return the hour of the time when the value of 'att1' was first equal to 50 in the past 24 hours.]

IF

Operator that introduces the condition in **IF-THEN-ELSE** statement. Return the first of two specified values if the conditional expression is true. Otherwise, return the second value.

Syntax

```
IF (expression) THEN (x) ELSE (y)
```

Arguments

- **expression**

  Any expression that evaluates to true or false

- **x, y**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return 'the value of 'att2'; otherwise return the value of 'att3']

IN

Return **True** if a value is included in a set or a range of inclusive values.

Syntax

```
x In (y1 [, ... yn])
```

```
x In (y1...yn)
```

Arguments

- **x**

  numeric value or string

- **y_1 [, ... y_n]**

  a set of numeric or string values

- **y_1 ... y_n**

  a range of numeric values

Returns

True when *x* is included in a set or a range of values

Exceptions

None

Example

- `1 In (9, 10)`

  [Returns `False` since 1 is not included in a set of 9 and 10]

- `2 In (1...10)`

  [Returns `True` since 2 is included in a range of 1 to 10]

InStr

Return the location within a string where a sub-string match is first found.

Syntax

```
InStr([start,] s1, s2 [,caseSensitive])
```

Arguments

- **start**

  Optional: An integer specifying which character in *s1* to start the comparison. Must be larger than or equal to 1.

- **s1, s2**

  Two strings to be compared.

- **`caseSensitive`**

  Optional: A flag indicating if the comparison is case-sensitive. If 0 (the default) the comparison is case-insensitive, if 1, the comparison is case-sensitive.

Returns

0 if *s2* is not a sub-string of *s1* starting from the start position; otherwise, the location of character where *s2* first matches the characters in *s1* from the start position.

Exceptions

Wildcard characters are not treated as wildcards.

Example

- `InStr("What", "At")`

  [Returns 3]

- `InStr("What What What", "What")`

  [Returns 1]

- `InStr("what", "At", 1)`

  [Returns 0]

- `InStr(4, "what", "At")`

  [Returns 0]

- `InStr('att1', "Error")`

  [Returns 1 if the value of *att1* is "Error"]

Int

Return the integer part of an integer or real number.

Syntax

```
Int(x)
```

Arguments

- **x**

  Number, Boolean, numeric string, timespan, or an attribute whose value evaluates to a number at time of evaluation

Returns

If *x* is a number, the integer part of *x* is returned. If *x* is a string, it is first converted into a number. If *x* is a *timespan*, the number of seconds from 12:00am January 1, 1970 is returned.

Exceptions

Returns error if the variable is not assigned

Notes

`Int('*')` returns UTC time in seconds from 12:00am January 1, 1970.

Example

- `Int('att1')`

  [Return the integer part of the value of 'att1' at current time.]

- `Int('*'-'t')`

  [Return how many seconds have passed since the start of day today.]

- `Int(2.1)`

  [Returns 2.]

- `Int("2.1")`

  [First converts the string to a number and returns 2.]

- `Int(true)`

  [Returns 1.]

InterpolatedValues

Obtain interpolated values over a specified time range using the specified time step.

Syntax

```
InterpolatedValues(attname, starttime, endtime, timestep)
```

Arguments

- **`attname`**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as '-3h') in reference to an absolute endtime

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as '+1h') in reference to an absolute starttime

- **`timestep`**

  a time interval enclosed in single quotes (such as '+1m' or '-10s') representing the incremental change in time between the specified range to obtain the interpolated values

Returns

An array of the values obtained within the specified range in intervals of the time step

Exceptions

If the attribute does not support range calls or interpolated values of range calls, the function returns an error indicating as such

Notes

- If the **`starttime`** is earlier than the **`endtime`**, the resulting values will be in time-ascending order, otherwise they will be in time-descending order.
- When a positive timestep is specified, the interval calculation begins at the earliest bounding time in the time range and applies the interval repeatedly in the time-ascending direction to generate the calculation intervals.
- If a negative timestep is specified, the interval calculation begins at the latest bounding time in the time range and applies the interval repeatedly in the time-descending direction to generate the calculation intervals.
- Note that the order of values returned will still be reflected by the time range, regardless of the timestep sign.

Example

- `InterpolatedValues('att1', 't', '+1h', '+10m')`

  [Return the values of 'att1' between 12:00 and 1:00am today in 10-minute intervals]

- `InterpolatedValues('att1', 't', '+1h', '-14m')`

  [Return the values of 'att1' between 12:00 and 1:00am today in 14-minute intervals, anchoring from 1:00am (that is, 12:04,12:18,12:32,12:46,1:00)]

IsSet

For an attribute associated with a PI point, determine if the point is annotated, substituted, or questionable.

Syntax

```
IsSet(attname, select)
```

Arguments

- **`attname`**

  An attribute associated with a PI point of value type integer, real number, enumeration value, or string

- **`select`**

  A string but only the first character is considered: "a" for annotate, "s" for substituted and "q" for questionable. Not case-sensitive.

Returns

`True` or `False`

Exceptions

None

Example

- `IsSet('att1', "a")`

  [Returns `True` if the value of 'att1' is annotated]

- `IsSet('att1', "s")`

  [Returns `True` if the value of 'att1' is substituted]]

- `IsSet('att1', "q")`

  [Returns `True` if the value of 'att1' is questionable]

LastValue

Get the last value in an array that satisfies a given condition.

Syntax

```
LastValue(array [, condition($val)])
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

The last value that satisfies the given condition

Exceptions

None

Notes

None

Example

- `LastValue(Data)`

  [Return the last value from an array named **`Data`**]

- `LastValue(Data, $val > 90 and $val < 105)`

  [From the array named **`Data`**, return the last value in the array that is greater than 90 and less than 105]

LCase

Convert a string to a lowercase string.

Syntax

```
LCase(s1)
```

Arguments

- **s1**

  string

Returns

A string that has been converted to lowercase

Exceptions

If the argument is not a string, returns an error value.

Example

- `LCase("String")`

  [Returns "string"]

Left

Return a specified number of characters of a string from the left.

Syntax

```
Left(s1, len)
```

Arguments

- **s1**

  String

- **len**

  Integer.

Returns

The first **len** characters of the string, starting from the left.

Exceptions

If the arguments are not of the required types, returns an error.

Example

- `Left("String_att", 3)`

  [Returns "Str"]

Len

Return the length of a string.

Syntax

```
Len(s1)
```

Arguments

- **s1**

  string

Returns

The length of a string

Exceptions

If the argument is not a string, returns an error value.

Example

- `Len("String")`

  [Returns 6]

- `Len('sinusoid')`

  [Returns `Calc Failed`]

LinRegr

Apply linear least squares fitting to two sets of values given by attributes over a specified time range. The output is a one-based array with the values of the slope, intercept and R^2.

Syntax

```
LinRegr(x, y, mode, starttime, endtime [, pctgood])
```

```
LinRegr(y, starttime, endtime [, pctgood])
```

Arguments

- **x**

  an attribute with the first set of time series data (such as PI point data reference) enclosed in single quotes

- **y**

  an attribute with the second set of time series data (such as PI point data reference) enclosed in single quotes

- **mode**

  a number that specifies how to align time-stamped values

  choose from 0, 1 or 2 where:

0 represents the combination of time-stamped values from *x* and *y*

1 represents values from both attributes according to *x*'s time stamps

2 represents values from both attributes according to *y*'s time stamps

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute *endtime*

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute *starttime*

- **`pctgood`**

  Optional. Minimum percentage of time during the time range that attribute values must be good.

  You can set *pctgood* as a threshold to ensure that there are sufficient good values to calculate **`LinRegr`**.

Returns

One-based array with the values of slope, intercept and R^2.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value

Notes

Bad values are excluded from **`LinRegr`** calculation

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **`PctGood`** function to find out what percentage of the values is good

Index the slope, intercept and R^2 outputs from [1].

Name	Expression	Value at Evaluation
RegressionData1	`LinRegr('XValues','YValues',2,'t','*')`	[0.27589, 219.05, 0.50575]
RegressionData2	`LinRegr('XValues','YValues',2,'t','*',80)`	[0.27589, 219.05, 0.50575]
Slope	`RegressionData1[1]`	0.27589
Intercept	`RegressionData1[2]`	219.05
RSquared	`RegressionData1[3]`	0.50575

Example

- `LinRegr('att1', 'att2', 0, 't', '+1h', 80)`

  [Return a one-based array of slope, intercept and R^2 based on time-stamped values of 'att1' and 'att2' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `LinRegr('att1', 'att2', 1, 't', '+1h')`

  [Return a one-based array of slope, intercept and R^2 based on time-stamped values of 'att1' between 12:00 and 1:00am today]

- `LinRegr('att1', 't', '+1h', 80)`

  [Return a one-based array of slope, intercept and R^2 of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `LinRegr('att1', 't', '+1h')`

  [Return a one-based array of slope, intercept and R^2 of 'att1' between 12:00 and 1:00am today]

Log

Return the natural logarithm (base e = 2.7182818...) of an integer or real number.

Syntax

```
Log(x)
```

Arguments

- **x**

  Must be an integer or real number greater than zero

Returns

The natural logarithm of *x*

Exceptions

If *x* is zero or negative, or not a number, returns an error value

Example

- `Log(14)`

  [Returns 2.6391]

- `Log(TagVal('att1', '14-Dec-16'))`

  [Return the natural log of the value of 'att1' at 12:00am on Dec 14, 2016]

Log10

Return the base 10 logarithm of an integer or real number.

Syntax

```
Log10(x)
```

Arguments

- **x**

  Must be an integer or real number greater than zero

Returns

The base 10 logarithm of *x*

Exceptions/Errors

If *x* is zero or negative, or not a number, returns an error value

Example

- `Log10(100)`

  [Returns 2]

- `Log10(TagVal('att1', '14-Dec-16'))`

  [Return the base 10 logarithm of the value of 'att1' at 12:00am on Dec 14, 2016]

Logbase

Return the logarithm of positive numeric value *x* to a specified base *y*.

Syntax

```
Logbase(x,y)
```

Arguments

- **x**

  An integer or real number greater than zero

- **y**

  A positive integer indicating the base of the logarithm

Returns

The logarithm of *x* to base *y*

Exceptions

If *x* is zero or negative, or not a number, returns an error value

Example

- `Logbase(256, 2)`

  [Returns 8]

- `Logbase(1000, 10)`

[Returns 3]

- `Logbase(TagVal('att1', '14-Dec-16'), 16)`

  [Return the logarithm of the value of 'att1' at 12:00am on Dec 14, 2016 to base 16]

LTrim

Remove the leading blanks from a string.

Syntax

```
LTrim(s1)
```

Arguments

- **`s1`**

  string

Returns

A string with leading blanks removed

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `LTrim(" String")`

  [Returns "String"]

- `LTrim("String ")`

  [Returns "String "]

MapData

Apply a function to each value in an array.

Syntax

```
MapData(array, function($val))
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`function($val)`**

  any supported PE statement, as a function of **`$val`**, that performs an operation. **`$val`** is a placeholder for each value in the array

Returns

The transformed array of all the values

Exceptions

None

Notes

None

Example

- `MapData(Data, (IF BadVal($val) THEN 0 ELSE $val))`

  [From an array named **Data**, return a new array with bad values set to 0]

Max

Return the maximum from a set of values.

Syntax

```
Max(x1, ... xn)
```

Arguments

- **x1, ... xn**

  Arguments or a single array of same value type (integers and real numbers, enumeration values, time expressions, or time intervals)

Returns

The maximum from a set of arguments.. The result is of the same data type as the arguments

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, **Max** returns an error. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs

Example

- `Max(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Find the maximum from these values: 14, value of 'att1' at trigger time, 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `Max('enum_att1', 'enum_att2', 'enum_att3')`

  [At trigger time, find the highest value from an enumeration set and return its name]

- `Max('*', 'y', 'Saturday')`

  [Find the most recent timestamp. Returns current time from the example above]

- `Max('*'-'*-1h', 't+8h'-'y+4h', '*'-'t')`

[Find the longest interval from these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `Max(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Max(Variable1)	9

[Find the maximum of values in an array `Variable1`]

Median

Return the median (middle value) of one or more values.

Syntax

```
Median(x1 [, ... xn])
```

Arguments

- **`x1 [, ... xn]`**

  Arguments and single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The median value of the input argument(s). If there are even number of arguments, the average of the two middle values is returned. Returns an error if the array does not consist of same value type.

Exceptions

Arguments whose run-time values are digital states are ignored. The function must have one or more arguments that evaluate to non-digital states; otherwise, `Median` returns an error value. Also returns an error if the array does not consist of same value type.

Notes

Arguments must be of same value type. The data type of the first argument sets the tone for the rest.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Examples

- `Median(14, 'att1', 14.5, TagVal('att2', '14-Dec-16'))`

[Find the median of these values: 14, the current value of *att1*, 14.5, and the value for *att2* at midnight on Dec 14, 2016]

- `Median('*', 'y', 'Saturday')`

  [Find the median of these timestamps: now, 12:00am yesterday, and 12:00am last Saturday]

- `Median('*'-'*-1h', 't+4h'-'y+8h', 'y+2h'-'20')`

  [Find the median of these time periods: from 1 hour ago to now, from 8:00am yesterday to 4:00am today, and 2:00am yesterday to the start of day (12am) on the 20th of this month]

- `Median(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Median(Variable1)	5

  [Find the median of values in an array `Variable1`]

Mid

Return a sub-string within a string.

Syntax

```
Mid(s1, start [,len])
```

Arguments

- **`s1`**

  string

- **`start`**

  An integer specifying the position of the first character within the string. The first character in the string is number 1.

- **`len`**

  Optional: The maximum length of the returned string. The default is the length of the string.

Returns

len characters of the string to the right of (and including) the first character whose position is specified by start.

Exceptions

If the arguments are not of the required types, an error value is returned. The maximum number of characters that can be returned is 999.

Example

- `Mid("String", 3)`

[Returns "ring"]

- `Mid("String", 3, 2)`

  [Returns "ri"]

Min

Return the minimum from a set of values.

Syntax

```
Min(x1, ... xn)
```

Arguments

- **`x1, ... xn`**

  Arguments or a single array of same value type (integers and real numbers, enumeration values, time expressions, or time intervals)

Returns

The minimum from a set of arguments. The result is of the same data type as arguments

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, **`Min`** returns an error. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs

Example

- `Min(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Find the minimum from these values: 14, value of 'att1' at trigger time, 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `Min('enum_att1', 'enum_att2', 'enum_att3')`

  [At trigger time, find the lowest value from an enumeration set and return its name]

- `Min('*', 'y', 'Saturday')`

  [Find the oldest timestamp from a set]

- `Min('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Find the shortest interval from these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `Min(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Min(Variable1)	1

[Find the minimum of values in an array `Variable1`]

Minute

Extract the minute from a time expression.

Syntax

```
Minute(t1)
```

Arguments

- **`t1`**

  A time expression, enclosed in single quotes.

Returns

The minute of time, in the range 0-59.

Exceptions

None.

Example

- `Minute('*')`

  [Extract the minute from current time.]

- `Minute(FindGT('att1', '-1h', '*', 5)`

  [Extract the minute from a timestamp when the value of 'att1' was first greater than 5 in the past hour. Return error if it was never over 5.]

Mod

The modulus operator (mod) returns the remainder from the quotient of two numeric values. For *x* **`Mod`** *y*, this is the remainder from *x*/*y*.

Syntax

```
x Mod y
```

Arguments

- **`x`**

  Any expression that evaluates to a numeric value

- **y**

  Any non-zero numeric value

Returns

Remainder from *x*/*y*

Exceptions

None

Notes

UOM of *x*, if it exists, is carried over to the result

Example

- `11 Mod 3`

  [Returns 2]

Month

Extract the month from a given time expression.

Syntax

```
Month(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The month of time, in the range 1-12.

Exceptions

None

Example

- `Month('*')`

  [Return the current month.]

- `Month(FindEq('att1', '-10d', '*', 5')`

  [Return the month from a timestamp when the value of 'att1' was first equal to 5 in the past 10 days.]

NextEvent

Find the timestamp of the next recorded value after a specified time.

Syntax

```
NextEvent(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The timestamp of the next recorded value after the specified time for the specified attribute

Exceptions

If there is no recorded value after the specified time, then the function returns `No Data`.

Example

- `NextEvent('att1', 't')`

  [Find and return the timestamp of the next recorded value of 'att1' since the start of day (12am) today]

NextVal

Find the next recorded value after a specified time.

Syntax

```
NextVal(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The next recorded value after the specified time for the specified attribute

Exceptions

If there is no recorded value after the specified time, the function returns an error.

Example

- `NextVal('att1', 't')`

  [Find and return the next recorded value of 'att1' since the start of day (12am) today]

Noon

Return a timestamp for noon on the day of a given time expression.

Syntax

```
Noon(t1)
```

Arguments

- **t1**

  A time expression enclosed in single quotes

Returns

A timestamp corresponding to noon of the day of the input time

Exceptions

None

Notes

This function is useful for establishing a unique clock time independent of the length of particular days.

Example

- `Noon('*')`

  [Return the timestamp for noon of the day when 'att1' was first equal to 50 in the past 3 days.]

- `Noon(FindEq('att1', '-3d', '*', 50))`

  [Return the timestamp for noon of the day when 'att1' was first equal to 50 in the past 3 days.]

NoOutput

Do not write current calculation result.

Syntax

```
NoOutput()
```

Arguments

None

Notes

It is important to include the parentheses after this function (use `NoOutput()` instead of `NoOutput` as `NoOutput` is an invalid syntax). This function applies only to the current calculation.

Example

- `If 'att1' < 100 or 'att1' > 200 then 'att1' else NoOutput()`

Normalrnd

Return a random number that maps the normal distribution curve using a specified mean and standard deviation.

Syntax

```
Normalrnd(x, y)
```

Arguments

- **x**

  A real number specifying the mean of the normal distribution curve

- **y**

  A real number specifying the standard deviation of the normal distribution curve

Returns

A random number that maps the normal distribution curve with a specified mean and standard deviation

Exceptions

None

Example

- `Normalrnd(300, 2.5)`

NOT

Return the negation of an expression. Return **True** if the truth value of an expression is false and **False** if the truth value of an expression is true.

Syntax

```
NOT expression
```

Arguments

- **expression**

  Any expression that evaluates to true or false

Returns

True when the expression is false and **False** when the expression is true

Exceptions

None

Example

- `NOT (1 In (1...10))`

  [Returns False since (1 In (1…10)) is true.]

NumOfChanges

Return the number of changes in value for an attribute within a specified time range.

Syntax

```
NumOfChanges(attname, starttime, endtime)
```

Arguments

- **attname**

  The name of an attribute.

- **starttime**

  A time expression that specifies the start of a time range, or a time span (such as '-3h') that specifies the start time relative to *endtime*; entries must be enclosed in single quotation marks

- **endtime**

  A time expression that specifies the end of a time range, or a time span (such as '+3h') that specifies the end time relative to *startime*; entries must be enclosed in single quotation marks

Returns

The count of changes in value for *attname* in the specified time range

Example

- `NumOfChanges('att1', 't', '*')`

  [Return the number of times the value of 'att1' changed since midnight until now]

OR

The logical disjunction of two expressions that returns **True** if either expression is true and **False** if both are false.

Syntax

```
expression1 OR expression2
```

Arguments

- **expression1**

  Any expression that evaluates to true or false

- **expression2**

  Any expression that evaluates to true or false

Returns

True if either expression is true and **False** if both are false

Exceptions

None

Example

- `('att1' > 50) OR ('att2' = "good")`

ParseTime

Translate a PI time expression to a timestamp. Use regular time expression inside single quotes for better performance.

Syntax

```
ParseTime(s1)
```

Arguments

- **s1**

  A character string or an array of strings in PI time format, enclosed in double quotes

Returns

The timestamp corresponding to **s1**

Exceptions

If **s1** is not a character string, or if there is a syntax error, returns an error value .

If the array does not consist of strings in a time format, returns an error.

Notes

Use of the time expression '01-Jan-2018' over the string "01-Jan-2018" is strongly recommended.

Example

- `ParseTime(Concat("12", "-31", "-16"))`

  [Returns 12/31/2016 12:00:00 AM, which is the same as '12/31/16']

- `ParseTime("14-Dec-16")`

  [Renders the same result as '14-Dec-16'. Use only when string operations are necessary]

- `ParseTime("*")`

  [Renders the same result as '*'. Use only when string operations are necessary]

- `ParseTime(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	'TimeStringArray'	[t, t+3h, t+14h]
Variable 2	ParseTime(Variable1)	[3/14/2018 12:00:00 AM, 3/14/2018 3:00:00 AM, 3/14/2018 2:00:00 PM]

[Renders the array results in a proper PI Time format, assuming today is 3/14/18]

PctGood

Find the time percentage, over a specified time range, for which the attribute had good values.

Syntax

```
PctGood(attname, starttime, endtime)
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

Returns

An integer or real number from 0 to 100: the percentage of the specified time for which the attribute had good values

Example

- `PctGood('att1', 't', '+1h')`

  [Return the time percentage between 12:00 and 1:00am today for which 'att1' had good values]

- `PctGood('att1', '-1h', '*')`

  [Return the time percentage between now and hour ago today for which 'att1' had good values]

Pi

Returns the value for pi.

Syntax

```
Pi()
```

Arguments

-

Returns

The value for pi

Exceptions

None.

Example

- `Pi()`

Poisson

Return a random number that maps a Poisson distribution with a specific mean.

Syntax

```
Poisson(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

A random number that maps a Poisson distribution with a specified mean

Exceptions

None

Example

- `Poisson(15)`

Poly

Return the polynomial $c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$.

Syntax

```
Poly(x, c0 [, ... cn])
```

Arguments

- **x**

  Variable. An integer or real number

- **c0 [, ... cn]**

  Coefficients. There must be at least one coefficient. All must be numbers.

Returns

The value of the polynomial

Exceptions

If *x* or any coefficient is not an integer or real number, **Poly** returns an error value

Example

- `Poly(3, 4, 5)`

  [Returns 19]

- `Poly('att1', 2, 3)`

Pow

Return **x** raised to the power **y**.

Syntax

```
Pow(x, y)
```

Arguments

- **x**

  numeric value

- **y**

  numeric value

Returns

The value of *x* raised to the power *y*

Exceptions

None

Notes

- The **Pow** function only works in PI Asset Framework (PI AF) analytics expression functions. Use caret (^) operator for same effect in PI Data Archive performance equations
- Both `Pow(2, 3)` and 2^3 are acceptable in PI AF analytics expression functions

Example

- `Pow(2, 3)`

  [Returns 8]

- `Pow(2.5, 3)`

  [Returns 15.625]

- `Pow(TagVal('att1', '*'), 8)`

  [Return the current value of 'att1' raised to the power 8]

PrevEvent

Find the timestamp of the last recorded value before a specified time.

Syntax

```
PrevEvent(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The timestamp of the last recorded value before the specified time for the specified attribute

Exceptions

If there is no recorded value before the specified time, the function returns an error.

Example

- `PrevEvent('att1', '*')`

  [Find the timestamp of the last recorded value of 'att1' before current time]

PrevVal

Find the last recorded value before a specified time.

Syntax

```
PrevVal(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The last recorded value before the specified time for the specified attribute

Exceptions

If there is no recorded value before the specified time, the function returns an error.

Example

- `PrevVal('att1', '*')`

  [Find and return the last recorded value of 'att1' before current time]

- `PrevVal('att1', '15-Mar-17')`

  [Find and return the last recorded value of 'att1' before start of day (12am) March 15th, 2017]

PStDev

Return the *population standard deviation* for a population of one or more values.

Syntax

```
PStDev(x1 [, ... xn])
```

Arguments

- **`x1, ... xn`**

  Arguments and a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The *population standard deviation* for the arguments. Returns a numeric value if the arguments are numbers. For arguments that are time expressions (time or time period), a number indicating a time period expressed in seconds is returned.

The population standard deviation of a population x1, ..., xn is

$$\sqrt{\frac{\sum (x_i - \mu)^2}{n}}$$

where μ is the mean of the arguments; that is,

$$\frac{\sum x_i}{n}$$

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, an error value is returned. Returns an error if the array does not consist of same value type.

Notes

`PStDev` uses every value in a population to calculate the population standard deviation. However, it is common, especially for a large population, to estimate standard deviation from a sample of the population. **`SStDev`** uses a set of sample values to calculate sample standard deviation, which approximates the population standard deviation.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `PStDev(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Return the "population standard deviation" of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `PStDev('*', 'y', 'Saturday')`

  [Return the "population standard deviation" of these timestamps.]

- `PStDev('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Return the "population standard deviation" of these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `PStDev(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	PStDev(Variable1)	2.8284

  [Find the "population standard deviation" of values in an array `Variable1`]

Rand

Return a random number between 0 and 1. For specified *x* and *y* values, return a random number between x - y/2 and x + y/2.

Syntax

```
Rand(x, y)
```

Arguments

- **x**

  A real number specifying the center point of the range

- **y**

  A real number specifying the size of the range.

If no arguments are specified, the default range is from 0 to 1.

Returns

A random number between 0 and 1. For specified *x* and *y* values, returns a random number between x- y/2 and x + y/2.

Exceptions

None

Example

- `Rand()`
- `Rand(500, 250)`

Range

Find the difference between the maximum and minimum values for an attribute during a specified time range.

Syntax

```
Range(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`pctgood`**

  Optional: Minimum time percentage, over the given time range, that values for the attribute must be good.

Returns

The difference between the attribute's maximum and minimum values during the specified time.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached in the given time range, an error is returned.

Notes

Bad values are excluded from **Range** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `Range('att1', 't', '+1h')`

  [Return the difference between the maximum and minimum values for 'att1' between 12:00 and 1:00am today.]

- `Range('att1', 't', '+1h', 80)`

  [Return the difference between the maximum and minimum values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

Rate

Return the difference over time between the current value and a previously evaluated value of an attribute or (time) expression.

Syntax

```
Rate(x)
```

Arguments

- **x**

  An attribute or (time) expression that resolves to a numeric value

Returns

- The calculated difference, using the value of the attribute, numeric, or timestamp value of the expression parameter, according to this formula: (current value – previously evaluated value) / (current timestamp – timestamp from previous evaluation).

Example

- `Rate(‘att1’)`
- `Rate('sinusoid')`
- `Rate(DaySec(‘*’) + 1)`

RecordedValues

Obtain values within a particular time range based on user-specified boundary type.

Syntax

```
RecordedValues(attname, starttime, endtime [, boundarytype])
```

Arguments

- **attname**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as '-3h') in reference to an absolute endtime

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as '+1h') in reference to an absolute starttime

- **`boundarytype`**

  a string indicating the data retrieval behavior at the end points of a specified time in double quotes:

  "Inside" - return the nearest recorded values inside the requested time range as the first and last values

  "Outside" - return the nearest recorded values on the outside of the requested time range as the first and last values

  "Interpolated" - create an interpolated value at the end points of the requested time range if a recorded value does not exist at that time

  > **Note:**
  > If unspecified, the default is the "Inside" mode.

Returns

An array of the values obtained within the specified range

Exceptions

If the attribute does not support range calls, the function returns an error with indication

Notes

- If the **`starttime`** is earlier than the **`endtime`**, the resulting values will be in time-ascending order, otherwise they will be in time-descending order.
- When no values are available but you specified "Outside" or "Interpolated" boundary types, a special value (**NoData**) is returned.
- This function retrieves up to 500,000 values at a time.

Example

- `RecordedValues('att1', 't+2h', 't+8h')`

  [Return the values of 'att1' between 2:00 and 8:00am today using the default "Inside" retrieval mode]

- `RecordedValues('att1', 't', '+1h', "Outside")`

  [Return the values of 'att1' between 12:00 and 1:00am today using the "Outside" retrieval mode]

RecordedValuesByCount

Obtain a specified number of values beginning at the requested start time in the direction specified.

Syntax

```
RecordedValuesByCount(attname, starttime, count [, timedirection, boundarytype])
```

Arguments

- **`attname`**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  a time expression representing the beginning of request

- **`count`**

  the number of stored (recorded) values to return. The value must be greater than zero

- **`timedirection`**

  a string indicating the time direction in data retrieval enclosed in double quotes:

  "Backward" - begin at the start time and move backward in time. Values will be returned in time-descending order

  "Forward" - begin at the start time and move forward in time. Values will be returned in time-ascending order

  > **Note:**
  > If unspecified, the default is the "Backward" mode.

- **`boundarytype`**

  a string indicating the data retrieval behavior at the end point of a specified time in double quotes:

  "Inside" - return the nearest recorded values inside the requested time boundary

  "Outside" - return the nearest recorded values on the outside of the requested time boundary

  "Interpolated" - create an interpolated value at the requested time boundary if a recorded value does not exist at that time

  > **Note:**
  > If unspecified, the default is the "Inside" mode.

Returns

An array of the specified number of values (if available)

Exceptions

If the attribute does not support the calls, the function returns an error with indication

Notes

- When no values are available but you specified "Outside" or "Interpolated" boundary types, a special value (**NoData**) is returned.

Example

- `RecordedValuesByCount('att1', 't+2h', 5)`

  [Return 5 values of 'att1' in time-descending order starting from 2:00am today using the default "Inside" retrieval mode moving backwards in time]

- `RecordedValuesByCount('att1', 't+8h', 3, "Forward", "Outside")`

  [Return the first 3 values of 'att1' since 8am today using the "Outside" retrieval mode]

Remainder

Return the remainder resulting from the division of x by y using the IEEERemainder method. This remainder is equal to x - (y Q), where Q is the quotient of x / y rounded to the nearest integer (if x / y falls halfway between two integers, the even integer is returned). If x - (y Q) is zero, the value +0 is returned if x is positive, or -0 if x is negative. If y = 0, NaN (not a number) is returned.

Syntax

```
Remainder(x, y)
```

Arguments

- **x**

  Any numeric value

- **y**

  Any non-zero numeric value

Returns

Return the remainder resulting from the division of x by y using the IEEERemainder method

> **Note:**
> This is not the same as the remainder returned using the Mod (modulus) operator

Exceptions

None

Notes

Unit of measure of *x*, if it exists, is carried over to the result

Example

- `Remainder(11, 3)`

  [Returns -1]

- `Remainder(10, 3)`

[Returns 1]

Right

Return a specified number of characters of a string from the right.

Syntax

```
Right(s1, len)
```

Arguments

- **s1**

  string

- **len**

  integer

Returns

len characters of the string from the right

Exceptions

If the arguments are not of the required types, an error value is returned.

Example

- `Right("String", 3)`

  [Returns "ing"]

- `Right("String", 20)`

  [Returns "String"]

Round

Round a number or time to the nearest unit.

Syntax

```
Round(x [, unit] [, roundingMode])
```

Arguments

- **x**

  Integer, real number, time expression, or time interval

- **unit**

  Optional. The size of the unit to round to. If *x* is a number, unit must be a number. If *x* is a time expression or time period, unit must be a time expression. If unit is omitted, **Round** rounds to the nearest even integer (for numbers) or second (for time expressions).

- **`roundingMode`**

  Optional. The type of rounding to apply. Using the string "AwayFromZero" applies rounding of midpoint values away from zero. Omission of this parameter, or the use of "ToEven", applies the default banker's rounding.

Returns

The nearest value to *x* which is an integer multiple of unit. Returns the same data type as *x*. For more details, see the following examples.

Exceptions

If *x* is a string, or if unit is of the wrong data type, an error value is returned.

Notes

The system uses a banker's rounding which rounds a number to the nearest even number.

If *x* is time and unit is omitted, this routine has no effect: times are accurate only to a second.

Unit of measure of the argument, if it exists, is carried over to the result.

Example

- `Round(12.499)`

  [Returns 12 (rounded to the nearest integer)]

- `Round(12.5)`

  [Returns 12 (rounded to the nearest even integer)]

- `Round(12.5, "AwayFromZero")`

  [Returns 13 (rounded up to the nearest integer)]

- `Round (13.5)`

  [Returns 14 (rounded to the nearest integer)]

- `Round(13.45, 0.1)`

  [Returns 13.4 (rounded to the nearest even number in one decimal place)]

- `Round(13.45, 0.1, "AwayFromZero")`

  [Returns 13.5 (rounded up to the nearest number in one decimal place)]

- `Round(1285, 10)`

  [Returns 1280 (rounded to the nearest even 10)]

- `Round(1285, 10, "AwayFromZero")`

  [Returns 1290]

- `Round(1285, 10, "ToEven")`

  [Returns 1280 (rounded to the nearest even 10)]

- `Round('14-Dec-16 11:47:16')`

  [Returns 14-Dec-16 11:47:16 (rounded to the nearest second)]

- `Round('14-Dec-16 12:30', '+1h')`

  [Returns 14-Dec-97 12:00:00 (rounded to the nearest even hour)]

- `Round('18:47:15'-'15:00:09')`

  [Returns 03:47:06 (rounded to the nearest second)]

- `Round('18:45:40'-'15:15:10','+1m')`

  [Returns 03:30:00 (rounded to the nearest minute)]

> **Note:**
>
> **Round** to the nearest day results in a timestamp of the closest day in UTC time and not local time.

Roundfrac

Round a numeric value x to y decimal places.

Syntax

```
Roundfrac(x, y [,roundingMode] )
```

Arguments

- **x**

  Must be an integer, real number or time expression

- **y**

  Integer that specifies the number of decimal places to round *x* to

- **roundingMode**

  Optional. The type of rounding to apply. Using the string "AwayFromZero" applies rounding of midpoint values away from zero. Omission of this parameter, or the use of "ToEven", applies the default banker's rounding

Returns

The value of x rounded to y decimal places. Returns the same data type as *x*. For more details, see the examples

Exceptions

If *x* is a string, or if unit is of the wrong data type, returns an error value

Notes

The system uses a banker's rounding which rounds a number to the nearest even number.

Unit of measure of the argument, if it exists, is carried over to the result.

If *x* is time and unit is omitted, this routine has no effect: times are accurate only to 1 second.

Example

- `Roundfrac (Pi(), 4)`

[Returns 3.1416]

- `Roundfrac (0.005, 2)`

  [Returns 0]

- `Roundfrac (0.005, 2, "ToEven")`

  [Returns 0.01]

- `Roundfrac (0.005, 2, "AwayFromZero")`

  [Returns 0.01]

- `Roundfrac (TagVal('att1'), 2)`

  [Return the value of 'att1' rounded to 2 decimal places]

RTrim

Trim trailing blanks from a string.

Syntax

```
RTrim(s1)
```

Arguments

- **s1**

  string

Returns

The source string with trailing blanks removed.

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `RTrim("String ")`

  [Returns "String"]

- `RTrim(" String")`

  [Returns " String"]

Sec

Return the trigonometric secant of a number.

Syntax

```
Sec(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The secant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sec(1)`

  [Returns 1.8508]

- `Sec(1.1)`

  [Returns 2.2046]

- `Sec('att1')`

  [Return the trigonometric secant of the value of 'att1' at time of evaluation]

Sech

Return the hyperbolic secant of a number.

Syntax

```
Sech(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic secant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sech(1)`

  [Returns 0.64805]

- `Sech(1.1)`

  [Returns 0.59933]

- `Sech('att1')`

  [Return the hyperbolic secant of the value of 'att1' at trigger time]

Second

Extract the second from a time expression.

Syntax

```
Second(t1)
```

Arguments

- **t1**

  A time expression enclosed in single quotes.

Returns

The second of time, in the range 0-59.

Exceptions

None.

Example

- `Second('*')`

  [Return the second from current time.]

- `Second(FindEq('att1', '-1m', '*', 2)`

  [Return the second from a timestamp when the value of 'att1' was first equal to 2 in the past minute.]

SecSinceChange

Return the time in seconds since an attribute value was updated.

Syntax

```
SecSinceChange(attname)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks.

Returns

The number of seconds since the last update of the value for an attribute.

Example

- `SecSinceChange('att1')`

  [Return the number of seconds since the value of 'att1' was updated]

Sgn

Return an indicator of the numerical sign of a number.

Syntax

```
Sgn(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

-1 if x < 0; 0 if x = 0; 1 if x > 0

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Sgn(1.1)`

  [Returns 1]

- `Sgn(0)`

  [Returns 0]

- `Sgn(-0.1)`

  [Returns -1]

- `Sgn('att1')`

  [Returns 1 if the value of 'att1' is positive, 0 if it equals 0, and -1 if it is negative]

Sin

Return the trigonometric sine of a number.

Syntax

```
Sin(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The sine of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sin(1)`

  [Returns 0.84147]

- `Sin(1.1)`

  [Returns 0.89121]

- `Sin('att1')`

  [Return the trigonometric sine of the value of 'att1' at trigger time]

Sinh

Return the hyperbolic sine of a number.

Syntax

```
Sinh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic sine of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sinh(1)`

  [Returns 1.1752]

- `Sinh(1.1)`

  [Returns 1.3356]

- `Sinh('att1')`

  [Return the hyperbolic sine of the value of 'att1' at trigger time]

Sqr

Return the square root of a number.

Syntax

```
Sqr(x)
```

Arguments

- **x**

  Must be an integer or real number equal to or greater than zero

Returns

The square root of *x*

Exceptions

If *x* is negative, or is not a number, returns an error value

Example

- `Sqr(2)`

  [Returns 1.4142]

- `Sqr(2.1)`

  [Returns 1.4491]

- `Sqr('att1')`

  [Return square root of the value of 'att1' at trigger time]

StateNo

Translate a digital state into its corresponding enumeration value.

Syntax

```
StateNo(digstate)
```

Arguments

- **digstate**

  An enumeration value

Returns

The offset into the Digital State Set corresponding to *digstate*

Exceptions

If a point is passed as *digstate* that is not a digital point, returns an error value

Notes

A digital state may appear more than once in the digital state table. In this case, the value that **StateNo** returns may vary. If *digstate* is the value of a digital point, **StateNo** returns a code number appropriate for that point

Example

- `StateNo(DigState("No Data"))`

  [Returns 248]

- `StateNo(TagVal('enum_att1', '*-1h'))`

  [Return the digital state number corresponding to the value of 'enum_att1' an hour ago]

SStDev

Return the *sample standard deviation* for two or more arguments that are a sample of a larger population.

Syntax

```
SStDev(x1, x2 [, ... xn])
```

Arguments

- **`x1, ... xn`**

  Arguments and a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The *sample standard deviation* of the arguments. Returns a numeric value if the arguments are numbers. For arguments that are time expressions (time or time period), a number indicating a time period expressed in seconds is returned.

The standard deviation of a sample x1, ... xn is equal to

$$\sqrt{\frac{\sum (x_i - \mu)^2}{n-1}}$$

Where μ is the sample mean; that is,

$$\frac{\sum x_i}{n}$$

Exceptions

Arguments whose run-time values are digital states are ignored. If there are not at least two numeric values, **`SStDev`** returns a zero. Returns an error if the array does not consist of same value type.

Notes

In the rare case where you have the entire population, rather than a sample, you might use the function **`PstDev`**, rather than **`SStDev`**.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `SStDev(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

[Return the "sample standard deviation" of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `SStDev('*', 'y', 'Saturday')`

  [Return the "sample standard deviation" of these timestamps.]

- `SStDev('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Return the "sample standard deviation" of these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `SStDev(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	SStDev(Variable1)	3.1623

  [Find the "sample standard deviation" of values in an array `Variable1`]

StDev

Find the time-weighted standard deviation of values for an attribute from a specified time range.

Syntax

```
StDev(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`pctgood`**

  Optional: Minimum percentage of time during the specified time range that attribute values must be good.

Returns

The time-weighted standard deviation for attribute values from the specified time range.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, an error is returned.

Notes

Bad values are excluded from **StDev** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `StDev('att1', 't', '+1h')`

  [Return the time-weighted standard deviation of values for 'att1' between 12:00 and 1:00am today.]

- `StDev('att1', 't', '+1h', 80)`

  [Return the time-weighted standard deviation of values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

String

Convert any value to a string.

Syntax

```
String(x)
```

Arguments

- **x**

  Any expression that is of normal PI value type

Returns

The string representing the value argument

Exceptions

None

Example

- `String("Hello, PI user!")`

  [Returns "Hello, PI user!"]

- `String(12.23)`

  [Returns "12.23"]

- `String('sinusoid')`

[Return current value of 'sinusoid' in string]

- `String('*')`

  [Return the current time in string]

TagAvg

Find the time-weighted average of values for an attribute during a specified time range.

Syntax

```
TagAvg(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of time during the time range that attribute values must be good.

Returns

The time-weighted average of attribute values during a specified time range.

Exceptions

If the point has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value.

Notes

Bad values are excluded from **TagAvg** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagAvg('att1', 't', '+1h')`

[Return the time-weighted average of values for 'att1' between 12:00 and 1:00am today.]

- `TagAvg('att1', 't', '+1h', 80)`

  [Return the time-weighted average of values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

TagBad

For an attribute associated with a PI point, test the point for an abnormal state at a specified time. If the point is an integer or real number, any digital state is considered abnormal. For digital points, the states that are defined for that point are normal; all others are abnormal.

Syntax

```
TagBad(attname [, t1])
```

Arguments

- **`attname`**

  An attribute associated with a PI point

- **`time`**

  Optional: A time expression. If omitted, current time ('*') is used.

Returns

`False` if the point's state at time is normal, `True` if it is abnormal.

Exceptions

None

Notes

`BadVal` can test any value or expression; **`TagBad`** can only test a PI point.

Example

- `TagBad('att1')`

  [Returns True if PI point for 'att1' does not exist or its value corresponds to system digital state (No Data, for example)]

- `TagBad('enum_att1', '14-Dec-16')`

  [Returns True if PI point for 'enum_att1' does not exist or its value corresponds to system digital state (No Data, for example) at 12:00am on December 14th, 2016]

TagDesc

For an attribute associated with a PI point, retrieve that point's descriptor from the point database.

Syntax

```
TagDesc(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's descriptor

Exceptions

If the PI point does not exist, an error value is returned

Example

- `TagDesc('sinusoid')`

  [Returns **12 Hour Sine Wave**]

- `TagDesc('cdt158')`

  [Returns **Atmospheric Tower OH Vapor**]

TagEU

For an attribute associated with a PI point, retrieve the point's engineering unit string from the point database.

Syntax

```
TagEU(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The PI point's engineering units

Exceptions

If the PI point does not exist, an error values is returned

Notes

Returns blank when PI point lacks an engineering unit

Example

- `TagEU('cdt158')`

  [Returns **DEG.C**]

TagExDesc

For an attribute associated with a PI point, retrieve the point's extended descriptor from the point database.

Syntax

```
TagExDesc(attname)
```

Arguments

- **attname**

  An attribute with a PI point data reference, enclosed in single quotes

Returns

The PI point's extended descriptor

Exceptions

If the PI point does not exist, an error values is returned

Notes

Returns blank when PI point lacks extended descriptor

Example

- `TagExDesc('cdt158')`

  [Returns blank since the point lacks extended descriptor]

TagMax

Find the maximum value of an attribute during a specified time range.

Syntax

```
TagMax(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of the time range that the attribute values must be good.

Returns

The attribute's maximum value during the specified time range.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, an error value is returned.

Notes

Bad values are excluded from **`TagMax`** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **`PctGood`** function to find out what percentage of the values is good.

- `TagMax('att1', 't', '+1h')`

  [Return the maximum value of 'att1' between 12:00 and 1:00am today.]

- `TagMax('att1', 't', '+1h', 80)`

  [Return the maximum value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

- `TagMax('att1', '14-Dec-16', '15-Dec-16')`

  [Return the maximum value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagMean

Find the average (mean) of values for an attribute during a specified time range.

Syntax

```
TagMean(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **pctgood**

  Optional: Minimum percentage of the specified time range that attribute values must be good.

Returns

The attribute's average value over the given time. Notice that the average is not time-weighted.

Exceptions

If the attribute has no good values or the **pctgood** minimum is not reached for the specified time range, an error value is returned. Unlike some other summary functions, **TagMean** does not interpolate any value on the boundary. Thus, if there is no value in the specified interval, an error value is returned.

Notes

Bad values are excluded from **TagMean** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagMean('att1', 't', '+1h')`

  [Return the average (mean) value of 'att1' between 12:00 and 1:00am today.]

- `TagMean('att1', 't', '+1h', 80)`

  [Return the average (mean) value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

- `TagMean('att1', '14-Dec-16', '15-Dec-16')`

  [Return the average (mean) value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagMin

Find the minimum value of an attribute during a specified time range.

Syntax

```
TagMin(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of the specified time range that the attribute values must be good.

Returns

The minimum value for the attribute during the specified time range.

Exceptions

If no good attribute values exist or if the *pctgood* minimum is not reached for the specified time range, an error value is returned.

Notes

Bad values are excluded from **TagMin** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagMin('att1', 't', '+1h')`

  [Return the minimum value of 'att1' between 12:00 and 1:00am today.]

- `TagMin('att1', 't', '+1h', 80)`

  [Return the minimum value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

- `TagMin('att1', '14-Dec-16', '15-Dec-16')`

  [Return the minimum value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagName

For an attribute associated with a PI point, get a point's name from the point database.

Syntax

```
TagName(attname)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The name of the PI point associated with *attname*

Exceptions

If the PI point does not exist, an error value is returned

Example

- `TagName('sinusoid')`

  [Returns **SINUSOID**]

- `TagName('cdt158')`

  [Returns **CDT158**]

TagNum

For an attribute associated with a PI point, get a point's number from the point database.

Syntax

```
TagNum(s1)
```

Arguments

- **`string`**

  The name of an attribute with a PI point data reference, enclosed in double quotes

Returns

The point's number

Exceptions

If the point does not exist, returns an error value

Example

- `TagNum("sinusoid")`

  [Returns 1]

- `TagNum("cdt158")`

  [Returns 3]

TagSource

For an attribute associated with a PI point, get a point's **point source** string from the point database.

Syntax

```
TagSource(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's **point source** string

Exceptions

If the point does not exist, an error value is returned

Example

- `TagSource('sinusoid')`

  [Returns R]

- `TagSource('cdt158')`

  [Returns R]

TagSpan

For an attribute associated with a PI point, get a point's span from the point database.

Syntax

```
TagSpan(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's span. If the point's type is **`digital`**, returns an integer whose value is the number of digital state defined for the point.

Example

- `TagSpan('sinusoid')`

  [Returns 100]

- `TagSpan('cdt158')`

  [Returns 200]

TagTot

Find the totalized value (time integral) of an attribute's values over a specified time range, optionally converting the totalized value from its current unit of measure (UOM) to a specified UOM in same class.

Syntax

```
TagTot(attname, starttime, endtime [, toUnit] [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **toUnit**

  Optional: The UOM to which the totalized value is converted. The specified UOM must be compatible with the UOM of the input attribute.

- **pctgood**

  Optional: Minimum percentage of the specified time range that the attribute's values must be good.

Returns

The totalized value (time integral) of an attribute's values over a specified time range.

Exceptions

- If the attribute has no good values or the **pctgood** minimum is not reached for the given time range, an error value is returned.
- If the specified UOM for conversion is incompatible with the UOM of the input attribute, an error value is returned. For example, if the UOM of the input attribute is gallons/minute, and the UOM for conversion is "m" (meters), then an error is returned.

Notes

- In case **toUnit** is not specified, the system chooses a scale factor such that the integral is correct only if the flow is expressed in units per day. If the flow is expressed in units per hour, or per some other time unit, you must multiply this result by a conversion factor. The conversion factor equals the number of actual flow time units in a day.

  For instance, if you totalize values measured in gallons per minute, multiply the result of **TagTot** by 1440 to get the answer in gallons. This conversion factor is not related to the time period you are totalizing over; it is strictly a function of the attribute's engineering units.

Some PI System installations may have the default total period configured to be something other than per day, such as per hour. Your conversion factor would be different in this case. Users are recommended to verify the installation standard used at their site.

- Bad values are excluded from **TagTot** calculation.
- When the percentage of good data is less than 100%, **TagTot** determines the total based on good data and divides the fraction of good data in the interval. The time period during which the data is bad is ignored when calculating the total.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the value is good.

Example

- `TagTot('att1', 't', '+1h')`

  [Return the totalized value of 'att1' between 12:00 and 1:00am today.]

- `TagTot('att1', 't', '+1h', "m")`

  [Return the totalized value of 'att1' calculated based on its unit between 12:00 and 1:00am today and convert to meters.]

- `TagTot('att1', 't', '+1h', 80)`

  [Return the totalized value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

- `TagTot('att1', 't', '+1h', "m", 80)`

  [Return the totalized value of 'att1' calculated based on its unit between 12:00 and 1:00am today and convert to meters when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

TagType

For an attribute associated with a PI point, get a point's **point type** from the point database.

Syntax

```
TagType(attname)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

Point type

Exceptions

If the point does not exist, returns an error

Example

- `TagType('sinusoid')`

  [Returns **Float32**]

- `TagType('cdm158')`

  [Returns **Digital**]

TagTypVal

For an attribute associated with a PI point, get a point's typical value from the point database.

Syntax

```
TagTypVal(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's typical value. If the point is an integer or real number, a number is returned; if the point's value type is digital, this indicates a digital state

Exceptions

If the point does not exist, returns an error

Example

- `TagTypVal('sinusoid')`

  [Returns 50]

- `TagTypVal('cdm158')`

  [Returns 3, a digital state]

TagVal

Return the value of an attribute at a specified time.

Syntax

```
TagVal(attname [, t1])
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks

- **`t1`**

  Optional: A time expression. If you omit this argument, '*' is used.

Returns

The value of an attribute at a specified time. This value is interpolated unless the associated PI Point has the Step property set to `True` or the value type of the attribute cannot be interpolated (for example, if it is a string)

Exceptions

If the associated point does not exist or has no value at the specified time, an error is returned.

Example

- `TagVal('att1')`

  [Return the value of 'att1' at current time]

- `TagVal('att1', 't+8h')`

  [Return the value of 'att1' at 8am today]

- `TagVal('enum_att1')`

  [Return the value of 'enum_att1', an attribute from an enumeration set at current time]

TagZero

For an attribute with a PI point data reference, get a PI point's "zero" value from the point database.

Syntax

```
TagZero(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's zero value. If the point type is integer or real number, this is a number; if the point's type is digital, this is a digital state (character string)

Exceptions

If the point does not exist, returns an error

Example

- `TagZero('sinusoid')`

  [Returns 0]

- `TagZero('cdt158')`

  [Returns 50]

Tan

Return the trigonometric tangent of a number.

Syntax

```
Tan(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The tangent of *x*.

Exceptions

If *x* is not a number, returns an error

Example

- `Tan(1)`

  [Returns 1.5574]

- `Tan(1.1)`

  [Returns 1.9648]

- `Tan('att1')`

  [Return the trigonometric tangent of the value of 'att1' at trigger time]

Tanh

Return the hyperbolic tangent of a number.

Syntax

```
Tanh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic tangent of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Tanh(1)`

[Returns 0.76159]

- `Tanh(1.1)`

  [Returns 0.8005]

- `Tanh('att1')`

  [Return the hyperbolic tangent of the value of 'att1' at trigger time]

Text

Concatenate strings representing argument values.

Syntax

```
Text(x1 [, x2, … xn])
```

Arguments

- **`x1 … xn`**

  Any expression of normal PI value type

Returns

A string that is the concatenation of strings representing the argument values.

Example

- `Text(Round('sinusoid', 1), " is the current value of 'sinusoid' at ", '*')`

  [Returns rounded current value of 'sinusoid' at current time]

THEN

Operator that returns the first of two specified values when the conditional expression in IF-THEN-ELSE statement is True.

Syntax

```
IF(expression) THEN (x) ELSE (y)
```

Arguments

- **`expression`**

  Any expression that evaluates to true or false

- **`x , y`**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return the value of 'att2'; otherwise return the value of 'att3']

TimeEq

Find the total time, within a range, when an attribute value is equal to a specified value. Time returned is in seconds.

Syntax

```
TimeEq(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`x`**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is equal to a given value.

Notes

Bad values are excluded from **`TimeEq`** calculation.

Example

- `TimeEq('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was equal to 80.]

- `TimeEq('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was equal to the value at 8am today. Return the result in seconds.]

- `TimeEq('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeGE

Find the total time, within a range, when an attribute value is greater than or equal to a specified value. Time returned is in seconds.

Syntax

```
TimeGE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is greater than or equal to a given value.

Exceptions

None

Notes

TimeGE interpolates between events, if necessary, to find the times when the attribute value crossed the given value.

Bad values are excluded from **TimeGE** calculation.

Example

- `TimeGE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was greater than or equal to 80.]

- `TimeGE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was greater than or equal to the value at 8am today. Result is in seconds.]

- `TimeGE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeGT

Find the total time, within a range, when an attribute value is greater than a specified value. Time returned is in seconds.

Syntax

```
TimeGT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is greater than a given value.

Exceptions

None

Notes

TimeGT interpolates between events, if necessary, to find the times when the attribute value crossed the specified value.

Bad values are excluded from **TimeGT** calculation.

Example

- `TimeGT('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was greater than 80.]

- `TimeGT('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was greater than the value at 8am today. Result is in seconds.]

- `TimeGT('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeLE

Find the total time, within a range, when an attribute value is less than or equal to a given value. Time returned is in seconds.

Syntax

```
TimeLE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is less than or equal to a given value.

Exceptions

None

Notes

TimeLE interpolates between archive events, if necessary, to find the times when the point crossed the given value.

Bad values are excluded from **TimeLE** calculation.

Examples

- `TimeLE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was less than or equal to 80.]

- `TimeLE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was less than or equal its value at 8am today. Result is in seconds.]

- `TimeLE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeLT

Find the total time, within a range, when an attribute value is less than a specified value. Time returned is in seconds.

Syntax

```
TimeLT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is less than a given value.

Exceptions

None

Notes

TimeLT interpolates between Archive events, if necessary, to find the times when the attribute value crossed the specified value.

Bad values are excluded from **TimeLT** calculation.

Example

- `TimeLT('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was less than 80.]

- `TimeLT('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was less than its value at 8am today. Result is in seconds.]

- `TimeLT('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeNE

Find the total time, within a range, when an attribute value is not equal to a specified value. Time returned is in seconds.

Syntax

```
TimeNE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is not equal to a specified value.

Exceptions

None

Notes

Bad values are excluded from **TimeNE** calculation.

Example

- `TimeNE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was not equal to 80.]

- `TimeNE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was not equal to the value at 8am today. Result is in seconds.]

- `TimeNE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeStamp

Return the time stamp for a single time-stamped value or an array of time-stamped values.

Syntax

```
TimeStamp(x)
```

Arguments

- **x**

  A time-stamped value or an array of time-stamped values

Returns

The timestamp of the specified time-stamped value

Exceptions

If the value of *x* is `No Data`, then `Calc Failed` is returned

Example

- `TimeStamp(PrevVal('sinusoid', '*'))`

  [Return the time stamp of the last recorded value before the current one for 'sinusoid']

- `TimeStamp(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5]
Variable 2	TimeStamp(Variable1)	[1/3/2018 8:15:00 AM, 1/3/ 8:16:17 AM, 1/3/2018 8:20:00 AM]

  [Assuming current time is 8:23 AM on 1/3/2018, returns the timestamps of 3 recorded values in the past 10 minutes]

Total

Return the sum of two or more values.

Syntax

```
Total(x1, x2 [, ... xn])
```

Arguments

- **x1, ... xn**

  Numbers, time intervals or a single array of numbers or time intervals

Returns

The total of the arguments. The result has the same data type as the arguments.

Exceptions

Arguments whose run-time values are digital states are not included in the total. If all values are digital states, **`Total`** returns an error value. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `Total('att1', 'att2', TagVal('att1', 'y+2h'), 40)`

  [Return the sum of following values: values of 'att1' and 'att2' at current time, the value of 'att1' at 2am yesterday and 40]

- `Total('t'-'y', '+1h')`

  [Returns 1.01:00:00]

- `Total(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Total(Variable1)	25

  [Return the sum of values in an array `Variable1`]

Trim

Trim blanks on both sides of a string.

Syntax

```
Trim(s1)
```

Arguments

- **`s1`**

  string

Returns

The source string with leading and trailing blanks removed.

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `Trim(" String ")`

[Returns "String"]

- `Trim(" String is a string attribute. ")`

  [Returns "String is a string attribute."]

Trunc

Truncate a number or time to the next lower unit.

Syntax

```
Trunc(x [, unit])
```

Arguments

- **x**

  An integer or real number, time expression, or time period.

- **unit**

  Optional. The size of the unit to truncate to; *x* will be truncated to a multiple of *unit*. If *x* is a number, *unit* must be a number. If *x* is a time expression or time period, *unit* must be a time period. If *unit* is omitted, **Trunc** truncates to the next lower integer (for a number) or second (for a time period).

Returns

The largest multiple of unit that is less than *x*. For a negative *x*, it returns the lowest multiple of *unit* larger than *x*. The return is the same data type as *x*.

Exceptions

If *x* is a string, or if unit is of the wrong data type, an error is returned.

Notes

If *x* is a time, and unit is omitted, this routine has no effect, as times are only accurate to one second.

When |x| < |unit|, 0 is returned.

Example

- `Trunc(12.999)`

  [Returns 12, truncated to the next lower integer]

- `Trunc(28.75, 10)`

  [Returns 20, truncated to next lower multiple of 10]

- `Trunc('14-Dec-16 11:47', '+1h')`

  [Returns 12/14/2016 11:00:00 AM, truncated to next lower hour]

- `Trunc('18:47'-'15:00','+1h')`

  [Returns 03:00:00, truncated period to next lower hour]

> **Note:**
> Truncating to the next lower day results in a timestamp of the next lower day in UTC time, not local time.

UCase

Convert a string to an uppercase string.

Syntax

```
UCase(s1)
```

Arguments

- **s1**

  String in double quotes

Returns

s1 in uppercase

Exceptions

If the argument is not a string, returns an error value.

Example

- `UCase("String")`

  [Returns "STRING"]

Weekday

Extract the day of the week from a given time expression.

Syntax

```
Weekday(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The day of the week, in the range 1-7, where 1 represents a Sunday.

Exceptions

None

Example

- `Weekday('*')`

[Return what day of the week today is]

- `Weekday(FindEq('att1', '-7d', '*', 50))`

  [Return what day of the week it was when the value of 'att1' was 50 for the first time in the past 7 days]

Year

Extract the year from a time expression.

Syntax

```
Year(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The year of time, in the range 1970-present

Exceptions

None

Example

- `Year('*')`

  [Return what year it is now]

- `Year(FindGT('att1', '1/1/1970', '*', 50))`

  [Return what year it was when the value of 'att1' was first greater than 50 since 1/1/1970]

Yearday

Extract the day of the year from a time expression. The day of the year (also known as a Julian day) is an integer ranging from 1 to 366, where 1 represents January 1.

Syntax

```
Yearday(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The day of the year of time, in the range 1-366, where 1 represents January 1

Exceptions

None

Example

- `Yearday('*')`

  [Return what day of the year today is, where 1 represents January 1]

- `Yearday('y')`

  [Return what day of the year yesterday was, where 1 represents January 1]

Steam functions for analysis expressions

You can include steam functions in analysis expressions to calculate the thermodynamic properties of steam. These functions are based on the 1997 IAPWS Industrial Formulation. The implementation is based on the combination of officially published forward and backward equations for all applicable regions.

> **Note:**
> Avoid confusing these steam functions with a similar set of steam functions used for PI Performance Equations in tag-based analytics. For more information on steam functions in asset and tag-based analytics, see Steam functions in asset and tag-based analytics.

The table below lists supported functions and descriptions. Ranges and units of measure for steam function properties are detailed in Ranges for steam function inputs. Reference states that apply to steam are listed in Steam property reference states.

Steam function descriptions

Function	Description
Steam_TsatP	Saturation temperature as a function of pressure.
Steam_HsatP	Saturation vapor specific enthalpy as a function of pressure.
Steam_SsatP	Saturation vapor specific entropy as a function of pressure.
Steam_VsatP	Saturation vapor specific volume as a function of pressure.
Steam_PsatT	Saturation pressure as a function of temperature.
Steam_HsatT	Saturation vapor specific enthalpy as a function of temperature.
Steam_SsatT	Saturation vapor specific entropy as a function of temperature.
Steam_VsatT	Saturation vapor specific volume as a function of temperature.
Steam_VPT	Vapor specific volume as a function of pressure and temperature.
Steam_VPTL	Liquid specific volume as a function of pressure and temperature. (For water.)
Steam_VPH	Vapor specific volume as a function of pressure and enthalpy.
Steam_VPS	Vapor specific volume as a function of pressure and entropy.
Steam_HPT	Vapor specific enthalpy as a function of pressure and temperature.
Steam_HPTL	Liquid specific enthalpy as a function of pressure and temperature. (For water.)
Steam_HPS	Vapor specific enthalpy as a function of pressure and entropy.
Steam_SPT	Vapor specific entropy as a function of pressure and temperature.
Steam_SPTL	Liquid specific entropy as a function of pressure and temperature. (For water.)

Function	Description
Steam_SPH	Vapor specific entropy as a function of pressure and enthalpy.
Steam_TPH	Temperature as a function of enthalpy and pressure.
Steam_TPS	Temperature as a function of entropy and pressure.
Steam_XPH	Steam quality (vapor fraction) as a function of pressure and enthalpy. (For wet steam.)
Steam_XPS	Steam quality (vapor fraction) as a function of entropy and pressure. (For wet steam.)
Steam_VPX	Steam specific volume as a function of pressure and steam quality. (For wet steam.)
Steam_HPX	Specific enthalpy as a function of pressure and steam quality. (For wet steam.)
Steam_SPX	Steam specific entropy as a function of pressure and steam quality. (For wet steam.)

Topics in this section

- Steam functions in asset and tag-based analytics
- Ranges for steam function inputs
- Unit of measure conversion for steam functions
- Steam property reference states
- Steam functions reference

Steam functions in asset and tag-based analytics

Asset and tag-based analytics tools both include steam functions that calculate thermodynamic properties of steam and water. Both sets of tools provide a complete set of steam functions but are based on different standards and have different implementations.

- **Steam functions in asset analytics**

  Steam functions in asset analytics are based on the 1997 IAPWS Industrial Formulation. The implementation is based on the combination of officially published forward and backward equations for all applicable regions.

  Note:
  The steam function names in asset analytics begin with "Steam_".

- **Steam functions in tag-based analytics**

  Steam functions in tag-based analytics refer to the PI PE Steam Functions Module (PI Steam), which is an extension to the PI Performance Equations Scheduler. These functions are based on the ASME Steam Tables, 6th Ed., and are also available in a COM library, `PISteamTableFunctions.dll`, which is distributed by other OSIsoft products, such as PI ACE.

  Note:
  Tag-based steam functions support both English and SI units; the function names begin with "StmSI_" or "StmEng_" for SI units and English units respectively.

Ranges for steam function inputs

The table below shows ranges and default units of measure for each of the five properties involved in steam function calculations.

The default units of measure are SI units. For information about using steam functions with other units of measure, see Unit of measure conversion for steam functions.

Pressure affects the state of steam. Several things to keep in mind for functions with pressure as an input are described in Considerations for steam functions that have pressure as an input.

Range of inputs for steam function properties

Function	Temp (°C)	Pressure (kPa)	Enthalpy(J/g)	Entropy(J/(g K))	Quality
Steam_TsatP		0.611212678 to 22064			
Steam_HsatP		0.611212678 to 22064			
Steam_SsatP		0.611212678 to 22064			
Steam_VsatP		0.611212678 to 22064			
Steam_PsatT	0.0 to 373.946				
Steam_HsatT	0.0 to 373.946				
Steam_SsatT	0.0 to 373.946				
Steam_VsatT	0.0 to 373.946				
Steam_VPT	0.0 to 800	0.611212678 to 100000			
Steam_VPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_VPTL	0.0 to 800	0.611212678 to 100000			
Steam_VPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_VPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_HPT	0.0 to 800	0.611212678 to 100000			
Steam_HPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_HPTL	0.0 to 800	0.611212678 to 100000			
Steam_HPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_SPT	0.0 to 800	0.611212678 to 100000			

Function	Temp (°C)	Pressure (kPa)	Enthalpy(J/g)	Entropy(J/(g K))	Quality
Steam_SPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_SPTL	0.0 to 800	0.611212678 to 100000			
Steam_SPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_TPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_TPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_XPH		0.611212678 to 22064	-0.04159 to 4160.7		
Steam_XPS		0.611212678 to 22064		-0.000155 to 11.921	
Steam_VPX		0.611212678 to 22064			0.0 to 1.0
Steam_HPX		0.611212678 to 22064			0.0 to 1.0
Steam_SPX		0.611212678 to 22064			0.0 to 1.0

Considerations for steam functions that have pressure as an input

Because pressure affects the state of steam, be aware of these considerations for functions that have pressure as an input.

Functions with pressure and temperature as inputs

Steam_HPT, **Steam_SPT** and **Steam_VPT** can be used for compressed water, superheated or saturated dry steam. For input temperature and pressure on the saturated curve, the calculated entropy, enthalpy or volume is a property of saturated vapor. These three functions can also accept high-range input temperatures (800 to 2000 °C) with pressures from 0.611212678 to 50000 kPa.

Steam_HPTL, **Steam_SPTL** and **Steam_VPTL** are used only for compressed water.

Functions with pressure and enthalpy or entropy as inputs

Steam_VPH, **Steam_VPS**, **Steam_HPS**, **Steam_SPH**, **Steam_TPH** and **Steam_TPS** can be used for compressed water, superheated or wet steam. For these functions, the range for enthalpy (H) is -0.04159 to 4160.7 J/g and for entropy (S) is -0.000155 to 11.921 J/(g K).

For **Steam_XPH** and **Steam_XPS**, input enthalpy or entropy greater than saturated vapor properties or less than saturated liquid properties returns an error.

Unit of measure conversion for steam functions

The default units of measure (UOM) for the steam functions are SI units. Any inputs to the steam functions defined in English units are automatically converted to the corresponding SI units. Similarly, if a steam function is configured with an output attribute, its output is automatically converted to the UOM configured for that attribute.

> **Note:**
> Automatic conversion requires that all UOM have been defined properly in the UOM database. If the definition of conversion from one unit to another of the same type is missing from the UOM database, conversion will fail.

In the expression editor, when you evaluate a steam function expression, results are displayed in the default SI units. To display the results in other UOM, you can use the **`Convert`** function.

The following example expressions use numeric values, variables, and attributes to show how steam functions handle UOM for input and output quantities.

Numeric input values

Numeric input values are recognized as quantities in the default UOM. In the following example for **`Steam_VPT`**, the numeric inputs are recognized as 300 kPa and 200 °C, and the output value is in cm^3/g, the default UOM for specific volume.

Name	Expression	Value	Output Attribute
`Variable1`	`Steam_VPT(300, 200)`	`716.44 cm3/g`	`Click to map`

Input and output attributes

In the following expression, **`Steam_VsatT`** takes as its only input the attribute 'TempSat', whose current value is 300 °F. The temperature is automatically converted to °C to evaluate the expression, and the result is displayed in the default SI UOM for **`Steam_VsatT`**.

Name	Expression	Value	Output Attribute
`Variable1`	`Steam_VsatT('TempSat')`	`403.7 cm3/g`	`OutputVolume`

The result is also mapped to the 'OutputVolume' output attribute. This attribute is configured in English UOM (ft^3/lb), so the result is automatically converted to that UOM. (On the Attributes tab in PI System Explorer, 'OutputVolume' would be listed with a value of 6.466627 ft^3/lb.)

Variables and the Convert function

This example uses variables and the **`Convert`** function to specify quantities in English UOM as steam function inputs.

1. In the P and T rows of the expression below, **`Convert`** assigns numeric values to English UOM for pressure and temperature.
2. In the B row, the variables P and T are automatically converted to SI units for the **`Steam_VPT`** calculation; the result is displayed in default SI UOM. The result is also mapped

to the output attribute 'OutputVolume'. That attribute is configured in English UOM (ft^3/lb), so the result is automatically converted to that UOM.

3. In the last row, **Convert** is used to convert the result (B) to English UOM and display it in the expression.

Name	Expression	Value	Output Attribute
P	Convert(2000, "psia")	2000 psia	Click to map
T	Convert(800, "°F")	800 °F	Click to map
B	Steam_VPT(P, T)	19.206 cm3/g	OutputVolume
Volume	Convert(B, "ft3/lb")	0.30762 ft3/lb	DisplayedVolume

Steam property reference states

The American Society of Mechanical Engineers (ASME) establishes the following reference states:

- **triple point**

  Triple point of water is at 273.16 K or 0.01 °C or 32.018 °F.

- **Celsius scale**

  The Celsius temperature is exactly T_k - 273.15.

- **critical point**

  Critical point of steam is at 647.096 K and 22,064 kPa, or 705.103 °F and 3200.1 psia.

- **reference state**

  The specific internal energy and specific entropy of the liquid phase were fixed at zero at the triple point of water.

Steam functions reference

Steam functions available for use in expression analyses are listed alphabetically.

Topics in this section

- Steam_HPS
  Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_HPT
  Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_HPTL
  Calculate the liquid specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_HPX
  Calculate the specific enthalpy as a function of pressure (**P** in kPa) and quality (vapor fraction).

- Steam_HsatP
  Calculate the saturated vapor specific enthalpy as a function of pressure (**P** in kPa).
- Steam_HsatT
  Calculate the saturated vapor specific enthalpy as a function of temperature (**T** in °C).
- Steam_PsatT
  Calculate the saturation pressure as a function of temperature (**T** in °C).
- Steam_SPH
  Calculate the vapor-specific entropy as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_SPT
  Calculate the vapor specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_SPTL
  Calculate the liquid specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_SPX
  Calculate the steam-specific entropy as a function of pressure (**P** in kPa) and quality (vapor fraction).
- Steam_SsatP
  Calculate the saturated vapor-specific entropy as a function of pressure (**P** in kPa).
- Steam_SsatT
  Calculate the saturated vapor-specific entropy as a function of temperature (**T** in °C).
- Steam_TPH
  Calculate the steam temperature as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_TPS
  Calculate the steam temperature as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_TsatP
  Calculate the saturation temperature as a function of pressure (**P** in kPa).
- Steam_VPH
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and enthalpy (**S** in J/g).
- Steam_VPS
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_VPT
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_VPTL
  Calculate the liquid-specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_VPX
  Calculate the steam-specific volume as a function of pressure (**P** in kPa) and quality (vapor fraction).

- Steam_VsatP
  Calculate the saturated vapor-specific volume as a function of pressure (**P** in kPa).
- Steam_VsatT
  Calculate the saturated vapor-specific volume as a function of temperature (**T** in °C).
- Steam_XPH
  Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_XPS
  Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).

Steam_HPS

Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed vapor specific enthalpy in J/g.

Example

- `Steam_HPS(2500, 7.5956)`

  [Returns 3684.9 J/g]

Steam_HPT

Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific enthalpy in J/g.

Example

- `Steam_HPT(40000, 600)`

  [Returns 3350.4 J/g]

Steam_HPTL

Calculate the liquid specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Only use for compressed water. For any **T** greater than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific enthalpy in J/g.

Example

- `Steam_HPTL(40000, 100)`

  [Returns 449.27 J/g]

Steam_HPX

Calculate the specific enthalpy as a function of pressure (**P** in kPa) and quality (vapor fraction). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed specific enthalpy in J/g.

Example

- `Steam_HPX(10000, 0.9)`

  [Returns 2593.7 J/g]

Steam_HsatP

Calculate the saturated vapor specific enthalpy as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed specific enthalpy for saturated vapor in J/g.

Example

- `Steam_HsatP(10000)`

  [Returns 2725.5 J/g]

Steam_HsatT

Calculate the saturated vapor specific enthalpy as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific enthalpy in J/g.

Example

- `Steam_HsatT(250)`

  [Returns 2801 J/g]

Steam_PsatT

Calculate the saturation pressure as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_PsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturation pressure of the steam in kPa.

Example

- `Steam_PsatT(250)`

  [Returns 3975.9 kPa]

Steam_SPH

Calculate the vapor-specific entropy as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed vapor specific entropy in J/(g K).

Example

- `Steam_SPH(10000, 3500)`

  [Returns 6.756 J/(g K)]

Steam_SPT

Calculate the vapor specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific entropy in J/(g K).

Example

- `Steam_SPT(40000, 600)`

  [Returns 6.017 J/(g K)]

Steam_SPTL

Calculate the liquid specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Only use for compressed water. For any **T** higher than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific entropy in J/(g K).

Example

- `Steam_SPTL(30000, 500)`

  [Returns 5.7956 J/(g K)]

Steam_SPX

Calculate the steam specific entropy as a function of pressure (**P** in kPa) and quality (vapor fraction). Use for only wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed specific entropy of the steam in J/(g K)

Example

- `Steam_SPX(15000, 0.7)`

  [Returns 4.8229 J/(g K)]

Steam_SsatP

Calculate the saturated vapor specific entropy as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed saturated vapor specific entropy in J/(g K)

Example

- `Steam_SsatP(10000)`

  [Returns 5.6159 J/(g K)]

Steam_SsatT

Calculate the saturated vapor specific entropy as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific entropy in J/(g K).

Example

- `Steam_SsatT(250)`

  [Returns 6.0722 J/(g K)]

Steam_TPH

Calculate the steam temperature as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed steam temperature in °C.

Example

- `Steam_TPH(40000, 3500)`

  [Returns 643.48 °C]

Steam_TPS

Calculate the steam temperature as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed steam temperature in °C.

Example

- `Steam_TPS(40000, 6)`

  [Returns 595.93 °C]

Steam_TsatP

Calculate the saturation temperature as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed saturation temperature in °C.

Example

- `Steam_TsatP(10000)`

  [Returns 311 °C]

Steam_VPH

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and enthalpy (**S** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed vapor specific volume in cm^3/g.

Example

- `Steam_VPH(25000, 3500)`

  [Returns 14.197 cm^3/g]

Steam_VPS

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed vapor specific volume in cm^3/g

Example

- `Steam_VPS(40000, 6)`

  [Returns 8.0055 cm^3/g]

Steam_VPT

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific volume in cm^3/g.

Example

- `Steam_VPT(50000, 500)`

  [Returns 3.8894 cm^3/g]

Steam_VPTL

Calculate the liquid specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C). For any **T** higher than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific volume in cm^3/g

Example

- `Steam_VPTL(75000, 500)`

  [Returns 2.308 cm^3/g]

Steam_VPX

Calculate the steam specific volume as a function of pressure (**P** in kPa) and quality (vapor fraction). Use for only wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed steam specific volume in cm^3/g

Example

- `Steam_VPX(15000, 0.7)`

  [Returns 7.7352cm^3/g]

Steam_VsatP

Calculate the saturated vapor specific volume as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed vapor specific volume in cm^3/g

Example

- `Steam_VsatP(10000)`

  [Returns 18.034 cm^3/g]

Steam_VsatT

Calculate the saturated vapor specific volume as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific volume in cm^3/g.

Example

- `Steam_VsatT(250)`

  [Returns 50.087 cm^3/g]

Steam_XPH

Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_XPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed steam quality (vapor fraction)

Example

- `Steam_XPH(10000, 2500)`

  [Returns 0.82888]

Steam_XPS

Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_XPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed steam quality (vapor fraction)

Example

- `Steam_XPS(10000, 5)`

  [Returns 0.72695]

Asset analytics native integration of external systems

MATLAB® Production Server™ is the first external system that asset analytics is integrating. By interfacing with the external system, users will be able to perform advanced calculations in expression and event frame generation analyses using the extended library of functions that reside outside of asset analytics.

Topics in this section

- Integration of MATLAB Production Server into asset analytics
- Configure an external system endpoint
- Supported functions and data types in MATLAB Production Server
- Error handling

Integration of MATLAB Production Server into asset analytics

Introduction

Beginning with the 2018 release, users of asset analytics can enjoy the extended capabilities of customized functions from MATLAB® Production Server™. Those who wish to make use of this functionality must procure and install MATLAB Production Server.

Requirements

Below is a simplified diagram depicting the connection and data transfer between asset analytics and MATLAB Production Server.

In order to take advantage of the extended features and functionality, users will need the following:

- PI Asset Framework 2018
- MATLAB Production Server 2018a with service discovery turned on. See the service discovery section in Supported functions and data types in MATLAB Production Server.
- MATLAB, MATLAB Compiler and MATLAB Compiler SDK
- Valid certificate if using the HTTPS schema for connection

Elements in the dotted box above must be set up before you can interface with the system at OSIsoft. For more information, refer to the MATLAB Production Server documentation (https://www.mathworks.com/help/mps/index.html).

Configure an external system endpoint

To configure MATLAB Production Server as an endpoint, see Configure an external system endpoint.

What to expect upon configuration

Once properly configured, you will be able to see all the available functions in the MATLAB Production Server shown within the functions list located on the right hand side of the Analyses tab. Expand **Functions** and select **MATLAB** from the drop-down list. Same as other asset analytics expression functions, function name, description, number of input and output arguments as well as value type will be displayed on the lower right-hand corner. You can click **Reload** to get the latest functions if you have made changes to them. Note that the automatic refresh occurs in the background periodically and this button is more for an immediate effect. If any functions found in the MPS are not supported, they will be greyed out and a tooltip will provide additional information.

Performance considerations

If you queue a lot of expensive high frequency calculations, it may negatively impact the overall performance of your system. We recommend that you use discretion and allocate resources accordingly. Avoiding frequent calls to the external system is also encouraged.

Configure an external system endpoint

Before you start

You must have the PI System Explorer Management plug-in installed before you can configure the external system endpoint. Make sure that you have a valid certificate if you are using the HTTPS scheme. Both HTTP and HTTPS are supported.

Procedure

1. Open PI System Explorer.
2. Go to **Tools** and select **External Systems**. Manage External Systems opens.
3. Enter the **Host** and **Port** information and click **Test Connection**. Make sure that you include a full host name including HTTP or HTTPS prefix. Upon successful connection, a list of available packages and the number of functions will be displayed.
4. Click **Save**.

Supported functions and data types in MATLAB Production Server

Supported functions

We only support functions with a proper function signature file. For each input and output argument, the type must be specified. See below for supported and unsupported data types.

Supported data types

The table here lists the MATLAB data types that asset analytics support:

MATLAB Data Type	Comment
Numeric (double, single, int8, uint8, int16, uint16, int32, uint32, int64, uint64)	Single value or one-dimensional array only
Char	
Logical (True, False)	
Cell	Single value or one-dimensional array only. As the string array is not a defined data type in MATLAB Production Server, we recommend using the cell array as an alternative.

Note:
If not specified in the function argument description, arrays are sent as row vectors from PI AF to the MATLAB Production Server.

Asset analytics will make best effort to convert the MATLAB Production Server data type into expected PI AF attribute type. In case the conversion fails, the function call will return a failure message.

With MATLAB Production Server 2018a, it is not yet possible to pass the value and the timestamp as a single argument from asset analytics. Therefore, this information needs to be sent on as two separate arguments to the MATLAB Production Server. In addition, the timestamp must be declared as data type double within the MATLAB code because the timestamp would be passed as POSIX time (also known as UNIX epoch time) when calling the MATLAB function.

In the following example, an array of values are retrieved from 'attribute1' within the specified time period by way of an expression function **`RecordedValues`** in asset analytics. Then the values and the timestamps are passed on to a fictitious function called MATLAB:CurveFitting.PolynomialFit, where MATLAB is the external system, CurveFitting is the name of the package and PolynomialFit is the function name.

Note:

Name	Expression
Values	RecordedValues('attribute1', 't', '*')
Variable1	MATLAB:CurveFitting.PolynomialFit(Values, TimeStamp(Values))

Within the MATLAB code, users must include a way to handle the timestamps that are passed on from asset analytics as POSIX time and convert it into a form that is consumable by MATLAB's native datetime object. This can be done in the following way, for example:

```
function calculated = PolynomialFit(values, timestamps)
timestampsConverted = datetime(timestamps,'ConvertFrom','posixtime')
```

For more information, see MATLAB documentation on datetime (https://www.mathworks.com/help/matlab/ref/datetime.html).

Note:
Time intervals in expression analyses performance equation syntax will be converted to seconds.

Data types not supported

Multi-dimensional array, struct and complex numbers are the MATLAB Production Server data types not supported in asset analytics.

Service discovery

The signature file in MATLAB Production Server is per package. The completeness of the signature is checked per function. This file must be added and/or modified each time a function is deployed.

Error handling

You will see relevant error messages surface in the UI to guide you.

There are 2 main types of errors: fatal and runtime. There are other errors not listed here.

- **Fatal errors**

  Analysis goes into an error as a result

  - Incorrect namespace of the MATLAB function
  - Incorrect package or a function name

  - Unsupported or incorrect number of arguments in a function
  - Invalid certificate if using HTTPS scheme

- **Runtime errors**

  Analysis does not go into error and the function evaluates to **Calc Failed**. Examples below are some common errors:

  - Divided by zero
  - Not a number when a number is required as an argument or as a MATLAB function output
  - Error in the MATLAB code; the error message will expose the type and location of the error
  - Connection is unavailable: server not available, expired MATLAB license and/or other HTTP/HTTPS errors
  - MATLAB function evaluation timeout: current setting is after 10 minutes

PI Analysis Service management

(These options are available if your installation of PI System Explorer includes the **Management** plug-in and you are signed in to a user account with permissions to configure PI Analysis Service.)

PI Analysis Service runs all analyses in the current PI AF server. Two features help you manage the service:

- The Analysis Service Statistics window enables you to explore statistics about the service to monitor its operations. These statistics are particularly important in diagnosing and identifying a solution for performance issues. You can save these statistics as a text file, which can provide a detailed snapshot of your analysis service to share with support staff helping you troubleshoot a problem. See View analysis service statistics.
- The Configuration window enables you to view or modify settings for analysis service properties to adjust performance. See View or modify analysis service.

View analysis service statistics

Procedure

1. In the navigator, click **Management**, and then select **Analysis** to view the analysis management area.
2. Right-click anywhere in the **Operations** panel and click **View Analysis Service Statistics**. The Analysis Service Statistics window displays groups of statistics you can use to monitor analysis service operations or to diagnose a performance issue and identify a solution.
3. Optional. Click **Refresh** to update statistics with current data.
4. Optional. Click **Save** to save current statistics as a text file. This enables you to share a snapshot of the current state of your analysis service with customer support staff.

View or modify analysis service

> **Note:**
> Changing these settings can greatly impact system performance. If you have any questions about changing the default settings, contact customer support staff for help.

Procedure

1. In the navigator, click **Management**, and then select **Analysis** to view the analysis management area.
2. Right-click anywhere in the **Operations** panel, and then click **Edit Analysis Service Configuration**.

   The Configuration window lists settings for analysis service properties. See Analysis service configuration for details about these settings.
3. To modify a property, edit its setting in the **Configuration** column.
4. Click **Ok** to apply any changes and close the window.

Analysis service configuration

You can modify these settings in the Configuration window. For information on how to configure the parameters below, see View or modify analysis service.

> **Note:**
> Changes to the **`AutoBackfillingEnabled`**, **`MaximumAllowedAutoBackfillingSpanInHours`** or **`NumberParallelDataPipes`** settings require a restart of PI Analysis Service to take effect.

- **AutoBackfillingEnabled**

  This property enables or disables automatic backfilling of gaps in data that result from periods when PI Analysis Service is not active. Automatic backfilling is enabled by default. In some cases, it may not be needed (in a test environment, for example). Enter `False` to disable this property. If you update this setting, you must restart PI Analysis Service.

  When an analysis is resumed after a short downtime, real-time calculations are started and, concurrently, auto-backfilling is also queued, if enabled. However, events with time-stamps before service start-time are processed differently depending on whether auto-backfilling is enabled using the **`AutoBackfillingEnabled`** configuration setting:

  - If auto-backfilling is enabled, triggered events with time-stamps before service start-time are processed by auto-backfilling and not by real-time processing.
  - If auto-backfilling is disabled, real-time processing will process all triggered events that are retrieved after the analysis service is started, including events that have time-stamps before analysis start-time.

- **AutoRecalculationEnabled**

  This property enables or disables automatic recalculation of analyses. Automatic recalculation executes analyses when a new input with a time stamp before the latest evaluated trigger is received. This useful feature automatically recalculates late-arriving inputs. Automatic recalculation is enabled by default. Enter `False` to disable this property.

> **Note:**
> Although automatic recalculation is enabled globally by setting this parameter, you need to opt in for an automatic recalculation at individual analysis (template) level. For more information, see Configure automatic recalculation of analyses and analyses templates.

- **AutoRecalculationIgnoreTimeInSeconds**

  This setting allows users to specify a duration whereby any input events with time stamps within this duration and older than the latest trigger time are ignored for automatic recalculation. The default value for this property is 30 seconds. Setting this parameter can be useful for filtering out noises in some analysis inputs that are not desirable for automatic recalculation, for example, if they are known to be delayed.

- **AutoRecalculationMinWaitTimeInSeconds**

  This setting controls how long analysis service will wait before queuing an analysis for automatic recalculation upon receiving an out-of-order triggering event. A valid range for this setting is 30 to 900 seconds, with a default of 60 seconds. The wait time is reset with the arrival of new out-of-order triggering events. Regardless of this setting, any analyses that have had an out-of-order triggering event will be queued for recalculation after 900 seconds.

- **CacheTimeSpanInMinutes**

  PI Analysis Service dynamically determines data cache time span for AF attributes used as inputs in calculations. For example, for analyses calculating hourly summary, PI Analysis Service would automatically cache an hour worth of data for required input attributes. This ensures that data required for calculations is available in the data cache, while unnecessary data can be trimmed. Manually set **`CacheTimeSpanInMinutes`** only if the time span cannot be determined by calculation configuration. A longer time setting means that any input data needed for calculations is available in the data cache, but it uses more system resources. For calculations that require earlier data (for example, from an hour earlier), you can specify a longer period for this setting to keep data cached, thus avoiding calls to PI Data Archive. The maximum time span that would be cached is 7 days and any data in the time span beyond 7 days would be trimmed. Note that the **`MaxCacheEventsPerAttribute`** setting takes precedence over this setting, so both settings may have to be adjusted simultaneously.

- **CalculationWaitTimeInSeconds**

  A period of time to wait before evaluating scheduled analyses. This wait time can be used to compensate for late-arriving inputs. For example, if a calculation depends on an interface with a known latency of up to 30 seconds, you might adjust this setting to at least 30 seconds to make sure you get the most current event from the interface. Longer settings reduce the chance of missing calculation inputs caused by latency but increase the delay in getting calculation results. Note that calculations use input values at the trigger time stamp when they are evaluated. The default value for this property is 5 seconds.

- **CreateFuturePIPointsForAmbiguousOutputTimes**

  A flag that determines the type of PI point to create for newly mapped output attributes that save output history when the time stamp of the output is uncertain. PI Analysis Service attempts to create the PI point type that matches the time stamp of the output, future PI points for output with future time stamps or historical PI points for output with present or past time stamps. If set to `False`, PI Analysis Service creates historical PI points in cases

when the time stamp of the output is ambiguous. If set to `True`, PI Analysis Service creates future PI points in cases when the time stamp of the output is ambiguous.

- **Default to Auto Create PI Points for Template**

  When this check box is selected, PI Analysis Service automatically creates PI points for output attributes in analyses derived from analysis templates.

- **EvaluationPartitionSize**

  Calculations are grouped for efficient evaluation. Analyses based on an analysis template with periodic scheduling, for example, are evaluated together as one calculation group. To speed up evaluation time, very large calculation groups are partitioned into subgroups based on the **`EvaluationPartitionSize`** setting; the subgroups can then be evaluated in parallel. The default value for this property is 10,000. For a group of analyses with long individual evaluation times, you may want to specify a smaller partition size. A group of 100 analyses, each performing summary calculations for a day's worth of data, may evaluate faster as 4 subgroups with a partition size of 25, depending on the CPU resources available on the machine.

- **EvaluationsToQueueBeforeSkipping**

  If it takes too long to execute a group of calculations, the queue of waiting calculations may become unmanageable. This setting determines when to skip calculations to avoid running out of resources. At the default setting, if more than 50 evaluations are found, PI Analysis Service starts skipping the oldest evaluations to try to return to its normal processing.

- **IsLoadSheddingEnabled**

  For some applications, accuracy and completeness are critical, and no calculations can be skipped regardless of the delay in completing them. To disable skipping calculations, enter `False` for this setting.

- **IsTelemetryAllowed**

  Enter `False` to stop sending usage data from PI Analysis Service for the Customer Experience Improvement Program. (This does not affect the corresponding setting for program participation you can configure in PI System Explorer.)

- **MaxAllowedAutoRecalculationSpanInDays**

  The maximum time span in days for which an analysis will be automatically recalculated on receiving an out of order event. Any recalculation time span exceeding this limit will not be considered for automatic recalculation. The default value for this parameter is 180 days.

- **MaxCacheEventsPerAttribute**

  Once this limit is reached, the oldest events are trimmed from the data cache. Note that this setting takes precedence over the **`CacheTimeSpanInMinutes`** setting.

- **MaxConcurrentRecalculationRequests**

  The maximum number of backfilling or recalculation operations that the PI Analysis Service can perform concurrently. PI Analysis Service dynamically scales the number of threads used for backfilling or recalculation between 1 and the number specified. This number is user configurable and depends on available resources (CPU and memory) of the machine running PI Analysis Service. You can set this parameter to any integer value between 1 and 4*P, where P represents the number of processors on the machine that PI Analysis Service is running. The default value for this parameter is (1/2)*P.

- **MaximumAllowedAutoBackfillingSpanInHours**

  Specify the maximum downtime (in hours) for the PI Analysis Service to initiate automatic backfilling. Setting the parameter to "0" or a negative number will restore the previously recorded setting for this option. If you update this setting, you must restart PI Analysis Service.

  > **Note:**
  > To disable automatic backfilling, you must disable the **`AutoBackfillingEnabled`** parameter.

- **MinCacheEventsPerAttribute**

  The default setting of 1 ensures that at least one event per attribute remains in the data cache.

- **NumberDataWriterThreads**

  The number of threads PI Analysis Service uses to write analysis outputs. More threads are useful for high frequency calculations that write values at high rates and for writing to PI Data Archive on a computer with high network latency (other threads can write values while waiting for a response from PI Data Archive). Consider increasing the number of threads when analyses skip many evaluations or analyses have high evaluation lag. More threads can help if data writes cause a bottleneck. You might set the number of threads to the number of processors on the computer where PI Analysis Service runs.

- **NumberEvaluationThreads**

  The number of threads PI Analysis Service uses for evaluating analyses concurrently. You can set this parameter to any integer value between 1 and 4*P, where P represents the number of processors on the machine that PI Analysis Service is running. The default value for this parameter is P. Consider increasing the number of evaluation threads when PI Analysis Service is not fully utilizing available CPU resources, for example, when analyses skip a number of evaluations or have high evaluation lag. This may happen when you have a lot of analyses with summary calculations that are triggered frequently. Having a larger number of evaluation threads is helpful for analyses that require frequent access to massive amount of input attribute data from outside the data cache.

- **NumberParallelDataPipes**

  Attributes receive updates from data sources, including PI Data Archive, through data pipes. Using a single data pipe to update a large number of attributes may take an unacceptably long time. You can increase the number of data pipes which can operate simultaneously to decrease update time. This can be especially useful for calculations with high-frequency data. Keep in mind, however, that increasing the number of data pipes increases the load on PI Data Archive. If you update this setting, you must restart PI Analysis Service.

- **RuntimeStorageFolderPath**

  Specify the location of the directory that contains PI Analysis Service run-time information such as user backfilling and calculation groups . By default, PI Analysis Service stores run-time information in `\\ProgramData\OSIsoft\PIAnalysisNotifications`.

- **Suggested PI Point Name**

  You can modify the naming pattern here with a substitution parameter. For a complete list, see List of PI AF substitution parameters. Note, however, that some parameters may not be meaningful in the context of a PI point name.

Buffering PI point outputs for PI Analysis Service

You can set up buffering for PI point attributes that write outputs to PI Data Archive from PI Analysis Service. This way, you can optimize throughput and create data resiliency by preventing data loss. You can access Buffering manager from PI System Explorer under **Tools** > **Buffering Manager**. See PI Server topics "Buffering" in Live Library (https://livelibrary.osisoft.com) and "Configure buffering for analysis outputs to PI points" in Live Library (https://livelibrary.osisoft.com) for more information.

Notifications

Notifications is the feature in PI Asset Framework (PI AF) that you use to create and manage notification rules. Notification rules are the mechanism by which users are alerted for response in real time, for conditions that signal events of interest. Notification messages may be customized to include information that is specifically relevant to the event, and may be sent via email to individuals, groups, or to a web service. Further, escalations are also configurable in the notification system.

Within PI System Explorer, you can configure and manage notification rules from the **Notification Rules** tab (visible after you select an element), and from the **Management** plug-in.

> **Note:**
> You can open the audit log directly from notification rules by right-clicking the notification rule and selecting **Audit Trail Events**. For more information on audit trail, see Track PI AF changes with Audit Trail.

Topics in this section

- Introduction to notifications
- Notifications and event frames
- Notifications history
- Requirements for notifications
- Client support for notifications
- Notifications security
- Create a notification rule
- Create a notification rule template
- Trigger criteria in notification rules
- Delivery formats in notification rules
- Subscriptions in notification rules
- Configuration of notifications delivery endpoints
- Contacts
- Configuration of notification rules for analyses or event frames
- Management of notification rules
- Changes from PI Notifications 2012
- Migration from PI Notifications 2012
- Additional resources

Introduction to notifications

Notification rules: Definition and scope

Notifications is included as a feature of PI Server, starting with the 2016 R2 version, and is not installed as a separate product. The notifications feature allows you to configure "notification rules" to send email messages or to call a web service; you can also configure message content and set up escalations.

> **Note:**
> Notifications created using PI Notifications 2012 or earlier are henceforth referred to as "legacy notifications".

Notifications are alerts that are generated when an event of interest is detected in the PI Server; such events are detected by examining and comparing real-time and static data with user-defined conditions configured in the PI Server. User-specified information can be configured in notifications that are sent to users or groups in real time to alert users of conditions to which they may need to respond. Notifications integrate with, and leverage, PI AF and other PI System services, and can be shared, managed, and maintained enterprise-wide. All notification actions, such as notification send times, acknowledgments, entry of comments, and escalations, are stored for later retrieval and examination.

Notification rules are based on event frames and notification messages can include information from event frame templates. For more information, see Notifications and event frames.

PI Notifications Service

PI Notifications Service is a Windows service that evaluates notification rules, processes incoming real-time event frames, and sends notifications. The PI Notifications Service need not run on the same machine as the PI Data Archive, PI AF server, or the client applications.

For information on event frames, see Event frames in PI AF.

Training video

For more information on using notifications in PI Asset Framework, watch the videos on the OSIsoft Learning channel playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2Jbcs43wQ2qO2dxW72PqGhZjKW&rel=0

Notifications and event frames

You can create notification rules based on any generated event frames (regardless of the method used to create them). The notification rules will be triggered when the configured condition or event is detected in the system or process, and a customized notification message is sent to a user, a group, or a web service.

Notification triggers are based on event frames; the event frames may come from different sources like analyses, the Event Frame Generator (EFGen) interface or custom AF SDK applications. Relevant information for events are stored in event frames and compatible tools can retrieve such information. Information contained in elements, event frame templates, and other places may also be added to the notification content. Further, event frame annotations contain historical notification information and may also include attachments; all annotations may be viewed using PI Vision or other client tools that support viewing annotations.

1. Analysis-generated event frames. For more information, see Event frame generation analyses to create analyses from the **Analyses** tab, and Trigger criteria in notification rules to create analyses from the **Notification Rules** tab of PI System Explorer.
2. Event Frame Generator (EFGen) interface-generated event frames. See Event Frame Generator documentation in Live Library (https://livelibrary.osisoft.com).
3. Custom AF SDK applications-generated event frames. See AF SDK documentation on the OSIsoft Tech Support website (https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/1a02af4c-1bec-4804-a9ef-3c7300f5e2fc.htm).
4. Manually created event frames.

Relationship between event frames and notification rules

Notifications history

Information associated with previous times that a notification was generated is stored as annotations on each event frame that triggered the notification. The annotations can be accessed by selecting the specific event frame of interest from the Event Frames plug-in on PI System Explorer. Each event frame annotation includes such information as the notification rule corresponding to the send event, the subscribers who were notified and when they were notified, and whether the attempt to notify was successful.

If you have migrated from PI Notifications 2012, and elected to migrate both configuration and history, your legacy notifications history is fully migrated; the historical information is migrated as annotations on the event frames created during migration. The migration report includes information on the legacy history instances that were processed during migration.

Note:
For information on permissions required during migration, see Migration from PI Notifications 2012.

Event frame annotations may also be viewed using PI Vision or other client tools that support viewing annotations.

Acknowledge a notification

In PI AF 2016, you can require that an event frame be acknowledged. The acknowledgement feature is used by notifications and PI AF client applications, such as PI Vision. The event-frame acknowledgement feature replaces the acknowledge notification functionality in legacy PI Notifications 2012 and older versions.

For more information on acknowledging event frames in PI AF, see Acknowledgement of event frames and Acknowledge event frames.

For more information on viewing and acknowledging event frames using PI Vision, see the PI Vision topic "View event details and annotate events" in Live Library (https://livelibrary.osisoft.com).

Requirements for notifications

Before you begin, make sure that you have installed PI AF Server 2017 R2.

To use the notifications feature, you must install, in this order:

1. PI AF Client 2017 R2
2. PI Notifications Service 2017 R2
3. Optional. PI Vision.
   - This is required if you want to use the acknowledgement link in the notifications feature, which is a link to the event details page in PI Vision.
   - To include a **Screenshot** from PI Vision in notifications, NTLM provider for Windows Authentication needs to be enabled in PI Vision. For more information, see Delivery formats in notification rules. Also see PI Vision topics "Enable Kerberos delegation using a custom PI Vision service account" in Live Library (https://livelibrary.osisoft.com) and "Enable Basic authentication" in Live Library (https://livelibrary.osisoft.com).
4. Management plug-in from PI AF Client 2017 R2 which allows for managing analyses and notification rules in bulk.

Client support for notifications

PI Vision, PI Coresight 2016 and later versions, and PI DataLink 2016 and later versions support notifications with the ability to display event frames and associated data. Notification messages can include links and screenshots to pre-configured PI Vision or PI Coresight displays, and thereby provide visual representation of the event that triggered the notification. See PI Vision, PI Coresight or PI DataLink product documentation for details on notifications feature support.

Notifications security

The notifications feature uses Windows security for communicating internally and with the PI AF server.

> **Note:**
> See the PI Server topic "PI AF objects and local permissions for PI Notifications Service" in Live Library (https://livelibrary.osisoft.com) for information on PI Notifications Service account permissions.

You can secure access to notification rules, notification rule templates, contacts, and contact templates. Object permissions are granted according to the PI AF Security model.

If Active Directory (AD) access is configured, information from the AD is accessed to create individual contacts. You cannot delete an AD contact from PI System Explorer, and you cannot modify security settings for the contact. To configure AD access, see the PI Server topic "Configure Active Directory objects for delegation" in Live Library (https://livelibrary.osisoft.com).

If you are using a screenshot feature in your email delivery format, see the installation topic "Set up a whitelist of trusted sites to take screenshots from" in Live Library (https://livelibrary.osisoft.com) on security guidelines. For more information, see also Add a screenshot in notification email delivery format and Add a domain to the whitelist.

In line with PI Asset Framework, Notifications supports Windows environments that enforce the use of U.S. Federal Information Processing Standards (FIPS) cryptographic algorithms. For more information, see **Support for FIPS** section in PI System Explorer and Microsoft Support article "System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" security setting effects in Windows XP and in later versions of Windows (https://support.microsoft.com/en-us/help/811833/-system-cryptography-use-fips-compliant-algorithms-for-encryption--has).

Topics in this section

- Security for contacts
- Security for notification rules and templates
- Configure authentication for SMTP server connection
- Configure authentication for a web service connection

Security for contacts

Microsoft Windows security is used to control access to groups, escalation teams, and custom, non-Active Directory contacts. Security settings for Active Directory contacts cannot be changed using PI System Explorer.

Newly created contact objects will inherit default permissions from the parent collections; a newly created custom contact will have security defined by the server's contacts collection.

Topics in this section

- Summary of permissions for contacts
- Edit security for existing contacts

Summary of permissions for contacts

Windows security must be set appropriately for contacts to view, modify, delete, or assign privileges to other contacts.

Permission	Description
`Read`	Can view information on contacts.
`Write`	Can modify contacts.
`Delete`	Can delete contacts.
`Admin`	Can assign privileges to contacts.

Edit security for existing contacts

Edit security settings from the **Contacts** plug-in to allow desired access to contacts. Administrator access is required to edit contact security. See Summary of permissions for contacts.

Note:
You can also edit security configurations for contacts by navigating to **Database** > **Edit Security**, and then selecting the appropriate **Notification Contact Templates** for your PI AF server (from the **Item** column). For example, if your PI AF Server is named MyServer, you must select **MyServer-Notification Contact Templates**.

Procedure

1. In PI System Explorer, click **Contacts** in the navigator pane.
2. Search for the group, escalation team or delivery endpoint whose permissions you want to change. You may need to double-click the contact to see its delivery endpoints.
3. Right-click the contact's delivery endpoint and select **Security**.

   The **Security Configuration** window for the delivery endpoint appears.
4. Change security settings for the delivery endpoint. Permissions can be changed for all identities that have been configured. You can also map new identities and configure their permissions.

Security for notification rules and templates

Notifications uses Microsoft Windows security to control access to notification rules and templates. To edit a notification rule or template, you need write access to notification rules.

Topics in this section

- Summary of permissions for notification rules and templates
- Edit security for a notification rule or template

Summary of permissions for notification rules and templates

Windows security must be set appropriately to perform specific tasks with notifications.

Permission	Description
`Read`	Can view a notification rule.
`Write`	Can modify settings in the Trigger Criteria, Subscriptions, and Formats panes.

Permission	Description
`Subscribe`	Can subscribe oneself to a notification and customize the subscription.
`SubscribeOthers`	Can subscribe others to a notification and customize their subscriptions.
`Delete`	Can delete a notification rule.
`Execute`	Can start, stop or reset a notification rule.
`Admin`	Can assign privileges to notification rules.

Write permission at the PI AF server level is needed for global formats. See Configure security for a PI AF server for more information.

Edit security for a notification rule or template

Edit security settings to allow desired access to notification rules and templates.

Note:

Administrator access is required to edit notification rule and template security. See Summary of permissions for notification rules and templates.

Procedure

1. In PI System Explorer, navigate to **Database** > **Edit Security**.

   You see the **Security Configuration** window.

2. From the **Item** column, select the appropriate notification rule or notification rule template for your PI AF server. For example, on a PI AF server named "MyServer", you must select:
   - **MyServer-Database-Notification Rule** to modify notification rule permissions
   - **MyServer-Database-Notification Rule Template** to modify notification rule template permissions.
3. Change security settings for notification rules or templates. Permissions can be changed for all identities that have been configured. You can also map new identities and configure their permissions.

Configure authentication for SMTP server connection

Best practice is to use a secure connection when using basic authentication. To use a secure connection with an SMTP server, enable the `Use TLS` option. The server must present a valid certificate for a secure connection to be established. To configure authentication for SMTP server connection, follow this procedure.

Procedure

1. To start PI System Explorer, click **Start** > **Programs** > **PI System** > **PI System Explorer**.
2. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
3. Click the **Plug-Ins** tab.

4. Scroll down to the **Delivery Channel Plug-Ins** section, then right-click **Email** and select **Settings**.
5. In the Email Delivery Channel Configuration window, click **Authentication Options** link under **SMTP Server**.

   Note:
   The same can be done for the Backup SMTP Server by clicking the link by the same name under it.

   Primary SMTP Server Authentication Options window opens.
6. Choose among the 3 authentication options.
   - Windows: PI Notifications Service will pass the network credentials of its service account to the SMTP server
   - Anonymous: no credentials will be provided to the SMTP server
   - Basic: enter a username and password the service will provide to the SMTP server

     Note:
     New username and password you put in will override existing credentials saved in the PI Notifications Service.
7. Click **Save and Close** or **Cancel**.
8. Optional. Check **Use TLS** box to enable encryption with basic authentication. Otherwise, the username and password will be sent in plain text.

Configure authentication for a web service connection

Best practice is to use a secure connection when using basic authentication. To use a secure connection for a web service, use the `https://` URI scheme. The server must present a valid certificate for a secure connection to be established. To configure authentication for a web service connection, follow this procedure.

Procedure

1. Go to **Contacts** plug-in.
2. Under **Delivery Endpoints**, select the web service delivery endpoint you want to authenticate.

   Note:
   - Both SOAP and REST delivery methods are supported.
   - Be sure to check in the delivery endpoint if it is newly configured.
3. Click a hyperlink next to **Authentication Option** at the bottom of the window. **Select Authentication Option** window opens.
4. Choose among the 3 authentication options.
   - Windows: PI Notifications Service will pass the network credentials of its service account to the web service
   - Anonymous: no credentials will be provided to the web service
   - Basic: enter a username and password the service will provide to the web service

> **Note:**
> New username and password you put in will override existing credentials saved in the PI Notifications Service.

5. Select **Basic** radio button.
6. Enter the user name and password and click **Save and Close**.

Create a notification rule

Configuring a notification rule includes specifying trigger criteria, adding subscribers to the notification rule, and formatting the message to suit the needs of your organization.

For a detailed sample, see Configuration of notification rules for analyses or event frames.

Procedure

1. In PI System Explorer, select the element on which you want to create notification rules.
2. From either the **Notifications Rules** tab, or from an existing event frame analysis, select **Create a new notification rule**.
3. Enter a name for the new notification rule and (optionally) select a category.
4. In the Trigger Criteria pane, specify the set of conditions that causes a notification to be sent. For more information, see Trigger criteria in notification rules.
5. In the Subscriptions pane, select **Manage Formats** and specify the format for notifications. For more information, see Delivery formats in notification rules.
6. In the Subscriptions pane, select **View/edit subscriptions** and specify the contacts to which notifications will be sent. For more information, see Subscriptions in notification rules.
7. Test that the notification is triggered when an event occurs that satisfies all of the trigger criteria specified.

Create a notification rule template

You can create a new notification rule template for an existing element template in PI System Explorer. You can also convert an existing notification rule into a notification rule template by right-clicking on the notification rule and selecting **Convert to Template**.

For a detailed sample, see Configuration of notification rules for analyses or event frames.

Procedure

1. In the navigator pane of PI System Explorer, click **Library**.
2. Select **Templates** > **Element Templates**.
3. Select the specific element template where you want to create the notification rule template.
4. Click on the **Notification Rule Templates** tab. The tab lists the notification rule templates already defined for the selected element template.
5. Create a new notification rule template for the selected element template. If no notification templates exist, click **Create a new notification rule template** to create the first one. Otherwise, click the New Notification Rule Template icon to create a new notification rule template.

6. Enter information to identify the notification rule template.
   - **Name**

     The name that uniquely identifies this notification rule template.
   - **Description**

     Optional text that describes the notification rule template.
   - **Categories**

     Optional category that you can assign to the notification rule template. Click the list and select the check box next to the category you want to assign, or click **New** to create a new category.
7. Configure trigger criteria for the notification rule template. For specific instructions, see Trigger criteria in notification rules.
8. Click on **Manage Formats** to customize the message that is sent with the notification. For specific instructions, see Delivery formats in notification rules.
9. Click on **View/Edit Subscriptions** to configure subscriptions. For specific instructions, see Subscriptions in notification rules.
10. Click **Check In**

to save the changes.

Trigger criteria in notification rules

You can configure trigger criteria for an event to determine when a notification will be sent to the appropriate delivery channel, for example, an email address or web service. The trigger conditions are configurable in the **Trigger Criteria** pane, and further options can be configured in the Options pane.

To see the Trigger Criteria and Options panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **View/Edit Trigger** in the Trigger pane.

Trigger Criteria pane

The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**.

Use analysis mode to trigger a notification rule on event frames generated by a particular analysis. In the **Analysis** mode, you can select an existing analysis or configure a new analysis of type **Event Frame Generation** or **SQC**, and optionally configure additional trigger conditions based on event frame attribute values. If you choose to create a new analysis with either the default or a custom name, you see the same analysis creation window as from the **Analyses** tab. For information on configuring an SQC analysis, see Create an SQC analysis. For information on creating an Event Frame Generation analysis, see Create an event frame generation analysis. For information on expression functions to be used as a trigger condition, see Expression functions reference.

Note:

- For notifications to take effect, you must check in both the analysis and the notification rule.
- When creating an SQC analysis from the **Trigger Criteria** pane, you can only choose **Event Frame** as the analysis output.

Use event frame search mode to trigger a notification rule based on event frame name, template and category. In the **Event Frame Search** mode, you can select a configured event frame template, from the drop-down list, and then configure the name and category for the event frames that will trigger your notifications. The name may contain wildcard characters that are supported by event frame search. You can optionally configure additional trigger conditions based on event frame attribute values.

For both **Analysis** and **Event Frame Search** modes, you can add additional trigger criteria using event frame attribute values. Criteria can be specified for any event frame template attribute that is specified in the notification rule trigger criteria. The attribute values that you select in the conditions must be specifically those for which you want to be notified of event frames (that match the analysis event frame template). For example, if your event frame template defines an event like "downtime" but you only want an email about "unplanned" downtime, you can configure an attribute value condition where a "reason code" attribute on the "downtime" event frame template has a value indicating "unplanned" downtime.

Note:
Multiple conditions are grouped using the `AND` operator.

Options pane

The Options pane allows you to configure these trigger options:

- **Resend Interval**

  The time interval after which PI Notifications Service will send additional alerts until the event frame matching the notification rule is acknowledged or closed.

- **Non-repetition Interval**

  The time interval during which PI Notifications Service will not send similar alerts associated with the same notification rule.

- **Event Frame Can Be Acknowledged**

  Option to enable event frame to be acknowledged; the event frame template is also modified accordingly. This option is automatically selected if the event frame template has been configured for acknowledgement.

- **Severity option**

  This option applies only to event frame generation analyses. If you have configured multiple start triggers for your analysis, you may choose to be notified in these ways:

  - When the current trigger severity is higher than any trigger severity encountered so far.
  - When the current trigger severity is higher than the previous trigger severity.
  - Every time a trigger condition is true, regardless of its relative severity to other previous triggers.

Topics in this section

- Non-Repetition Interval in notification rules
- Resend Interval in notification rules

Non-Repetition Interval in notification rules

You can configure the **Non-Repetition Interval** setting to prevent notifications from sending similar alerts, which are associated with the same condition, within a specified amount of time.

Alerts will be sent for triggering events with a higher severity even if a prior alert has been sent for a triggering event of lower severity, and within the time-frame of the non-repetition interval.

The system determines when to resend the notification based on the time the last notification was sent, and not based on the start time of the triggering event. For example, if the event frame started at time `t`, and the service, after processing, sends the notification at time `t1`, the system uses `t1` as the basis to calculate the time between alerts.

> **Note:**
> If you specify both a resend interval and a non-repetition interval, the resend interval must be longer.

Resend Interval in notification rules

You can configure the **Resend Interval** setting so that additional alerts are sent when the event frame has not been acknowledged or closed; it is a reminder that the event of interest leading to the notification is yet to be resolved. The resend interval specifies the time between sending successive alerts.

The system determines when to resend the notification based on the time the last notification was sent, and not based on the start time of the triggering event. For example, if the event frame started at time `t`, and the service, after processing, sends the notification at time `t1`, the system uses `t1` as the basis to calculate the time between alerts.

> **Note:**
> If you specify both a resend interval and a non-repetition interval, the resend interval must be longer.

Delivery formats in notification rules

Message and **Content** panes in PI System Explorer

From the Message and Content panes, you can configure the format of the message that is delivered when a notification rule is triggered.

To see the Message and Content panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **Manage Formats** in the **Subscriptions** pane.

Global default formats

The system provides one global format that is used as the default format when a subscriber is added to a notification rule. You can edit or rename the global format, but you cannot delete it.

You can also create additional global formats. These global formats can be viewed in notification rules or notification rule templates. To create, edit or delete global formats, navigate to **Tools** > **Global Formats** in PI System Explorer. For more information on global configuration parameters, see the OSIsoft Tech Support article KB01630 - Manually Managing Global Configuration Parameters in Notifications 2.x (http://techsupport.osisoft.com/Troubleshooting/KB/KB01630/).

Custom delivery formats

To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon. The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names.

To customize your delivery format, you can drag and drop the following types of information from the Content pane to the Message pane:

- PI AF Server properties
- Database properties
- Notification Rule properties
- Event frame properties
- Event frame attributes. If an event frame template is selected, you may add the event frame template attributes to the content of your email notification.
- Referenced element properties
- Referenced element attributes

These properties and attributes can provide a lot of input to analyzing the event of interest; for example, including the event frame start time property can tell you when the event of interest started. Including specific event frame and referenced element attributes can give you supporting information about key parameters.

For PI Vision users, notifications in PI AF 2017 R2 gives the option to construct a link that specifies a full path to a display. You can select **Full Path** from **Referenced element properties** or **Referenced element attributes** under the Content pane.

Note:

- PI Notifications Service and PI System Explorer 2017 R2 or later are required to use the **Full Path** option.
- **Full Path** can also be configured for global delivery formats.
- The existing **Path** option is relative to the reference element or attributes.

When sending a notification email, you can include a web, PI Vision URI, or a file link. That way, you can easily navigate to specific locations for relevant information about attributes, event frames, and so on. To add a link, click in the Message pane, and then click the link icon. You will be prompted to enter your base address. When you click the **Continue** button, **Create a link** window will open. There, you can configure your link.

Notifications in PI AF 2017 R2 has a new link-related feature. You now have an option to include a **Screenshot** (or a static image) of the data that triggered the notification. This way,

separate data retrieval from the source is not necessary, which would have been the case if only a URI (link) was included in the email. You can also specify display options if you choose to display the link as a screenshot. See Add a screenshot in notification email delivery format.

Note:

- Screenshots should come from a trusted source. For more information, see Add a domain to the whitelist.
- To add a **Screenshot** from PI Vision, make sure that NTLM provider is enabled for Windows Authentication in PI Vision. For more information on system requirements, see Requirements for notifications.

You can edit the message if you choose to display your link as a text. **Display link as** option can be found in 2 other places: in **Event Details Hyperlink** (if PI Vision is configured) and URI hyperlink (under **Referenced element attributes**; see Create URI Builder data references for more information).

If you are configuring a Web URI, you can specify a **fragment**, which is a string that is appended at the end of the URI. For more information on query string parameters, see the PI Vision topic "URL parameters reference" in Live Library (https://livelibrary.osisoft.com).

You can add a file attachment to your delivery format by clicking on the **+** symbol in the **Attachments** field. The selected attachment will be uploaded as an attribute on the referenced element. For acceptable types of attachment files, see `/FileExtensions` in AFDiag utility parameters.

Tip:
If the Content pane has a message indicating that your selected format is "read-only", you must first add a new format before you can customize it.

Training video

For more information on message formats, watch the OSIsoft Learning channel video:

https://www.youtube.com/watch?v=2q3YBcaNlEM&rel=0

Add a domain to the whitelist

Protect your system from unwanted security incidents by allowing screenshots only from a trusted source. To add a website to the trusted-sites list, follow this procedure.

Procedure

1. Click **Database** on the PI System Explorer toolbar.
2. Select **Configuration** database.
3. Click **OSIsoft** > **PIANO** in the Elements pane.
4. Go to **Attributes** tab and select **PageRender** > **TrustedSites**.
5. Add the sites in the **Value** field of the attribute delimited by space. Wildcard (*) character is allowed. Here are some examples:

   Note:
   Whitelisting applies to the website and all of its embedded elements (images, scripts).

If you entered ...	The system allows ...	The system blocks ...
*	All websites	No website
https://*	All websites using the https://URI scheme	All other websites
https://*.osisoft.com	https://www.osisoft.com https://techsupport.osisoft.com https://www.osisoft.com/index.html https://www.osisoft.com:80/index.html	http://www.osisoft.com www.osisoft.com https://osisoft.com
www.osisoft.com/index.html	www.osisoft.com/index.html www.osisoft.com/index.html?abc=123	www.osisoft.com/index www.osisoft.com/index.htmlabc
www.osisoft.com/abc	www.osisoft.com/abc www.osisoft.com/abc/def www.osisoft.com/abc.html www.osisoft.com/abc?def=123	http://www.osisoft.com/abcdef www.osisoft.com/abc.def
*.osisoft.com	www.osisoft.com www.osisoft.com:80 www.osisoft.com/index.html techsupport.osisoft.com https://www.osisoft.com http://www.osisoft.com/index.html	osisoft.com notosisoft.com www.osisoft.com.baddomain
osisoft.com	www.osisoft.com www.osisoft.com:80 www.osisoft.com/index.html techsupport.osisoft.com https://www.osisoft.com http://www.osisoft.com/index.html	notosisoft.com osisoft.com.baddomain
https://localserver/PIVision	https://localserver/PIVision https://localserver/PIVision/#/Displays	https://localserver/ https://localserver/PI http://localserver/PIVision https://localserver.int/PIVision

If you entered ...	The system allows ...	The system blocks ...

`*://*.int`	`http://internal.website.int`
	`http://internalwebsite.int`
	`http://internal.website.int:80`
	`http://internal.website.int/index.html`
		`https://www.osisoft.com`
		`http://www.osisoft.com`

`192.168.1.*`	`192.168.1.123`
	`192.168.1.123/abc.html`
	`http://192.168.1.123:8080/abc.html`
		`192.168.22.123`

Add a screenshot in notification email delivery format

Notifications in PI AF 2017 R2 has a new link-related feature. You now have an option to include a **Screenshot** (or a static image) of the data that triggered the notification. To add a screenshot in notification email delivery format, follow this procedure. For more information on security, see Add a domain to the whitelist and installation topic "Set up a whitelist of trusted sites to take screenshots from" in Live Library (https://livelibrary.osisoft.com).

Procedure

1. Navigate to the element that you want to receive the notification.
2. Click on the **Notification Rules** tab.
3. Select the notification rule that you want to add a screenshot. Click **Create a new notification rule** if you are creating a new notification rule.
4. Click **Manage Formats** in the Subscriptions pane.
5. Copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon .
6. Click in the Message pane, and then click the link icon.
7. Enter your base address in Paste a Link window and click **Continue**.
8. In the **Display link as** row, click the **Screenshot** radio button.
9. Optional. In the **Image options** row, choose **Format** and **Size** from each drop down list.
10. Click **OK**.

Subscriptions in notification rules

Contacts and Subscriptions panes in PI System Explorer

From the Subscriptions pane, you can configure the subscribers to be alerted when a notification rule is triggered. A subscriber is an individual entity or group of entities that can receive notification messages by subscribing to notification rules.

To see the Contacts and Subscriptions panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **View/Edit Subscriptions** in the Subscriptions pane.

To add a subscriber from the **Notifications Rules** tab of PI System Explorer, simply drag and drop a contact from the Contacts pane to the Subscriptions pane.

> **Note:**
> To unsubscribe a contact, right-click on the contact and select **Unsubscribe**.

When you install PI AF, information from Active Directory (AD) is imported and individual contacts are automatically created in Contacts. For more information, see Configure Active Directory access for contacts. If a user does not have an AD account, you can create a custom contact.

Subscriptions pane

When a contact is added to the Subscriptions pane, the following details are included for the contact:

- **Name**

  The name of the contact.

- **Configuration**

  The default, inherited, or custom delivery format for this subscriber.

- **Notify Option**

  The notify option may be one of:

  - Event start and end. You can receive a notification when an event frame matching the notification rule is created or closed.
  - Event start.
  - Event end. You can receive a notification only when an event frame is closed.

Contacts pane

Subscribers may be configured from these types of contacts on the Contacts pane:

- Individual contact
- Escalation teams
- Groups
- Delivery endpoints

Use **Contacts Search** to find individual contacts to include in a group or escalation team, or subscribe to a notification rule. You may use wildcards to expand your name searches; this will search both first and last names. For example, searching for "A*" will list both "Adam Smith" and "John Adams".

> **Note:**
> The search results show the display name as configured in Active Directory.

For more information on configuring contacts using the Contacts plug-in, see Contacts.

Training video

For information on how to configure subscribers, watch this video:

https://www.youtube.com/watch?v=lN2r5u5_U2w&rel=0

Customization of subscription content and delivery

The default delivery format specified on the **Message** tab is automatically applied to all subscriptions in the notification. To send a different message, you can customize subscription formats by applying a different delivery format.

> **Note:**
>
> If you customize subscriptions and later want to change delivery formats, you might need to manually change the delivery formats of subscriptions that you previously customized. For example, if you change the default delivery format, or specify a different delivery format as the default, only subscriptions that inherit that format will be changed; customized subscriptions are not affected.

For more information on delivery formats, see Delivery formats in notification rules.

Configuration of notifications delivery endpoints

A delivery endpoint is a single entity to which a notification is delivered, such as an individual email address or web service call. Each delivery endpoint is associated with a delivery channel, the conduit through which notification messages are sent. Notifications provides delivery channels for email and for both SOAP and REST web services.

To configure email delivery endpoints, see Configure an email delivery endpoint. To configure dynamic email delivery endpoints, new in PI AF 2017 R2, see Configure a dynamic email delivery endpoint.

To configure web service delivery endpoints, see Configure a REST web service delivery endpoint.

To configure notifications to be sent as text messages, see the OSIsoft Tech Support article KB00294 - How do I set up PI Notifications to send text messages to my cellular phone (https://techsupport.osisoft.com/Troubleshooting/KB/KB00294).

Topics in this section

- Configure an email delivery endpoint
- Configure a dynamic email delivery endpoint
- Configure a REST web service delivery endpoint

- Configure a SOAP web service delivery endpoint

Configure an email delivery endpoint

To configure an email delivery endpoint and add an email subscriber to your notification rule, follow this procedure.

Before you start

If you have not already configured the SMTP server settings for the email delivery channel, follow the instructions in: Update SMTP settings for the email delivery channel.

Procedure

1. Add an email delivery endpoint using the PI System Explorer Contacts plug-in. For specific information, see Manage contacts and Options for the notifications email delivery channel. See Configure authentication for SMTP server connection for setting up authentication.
2. Navigate to **Elements** > ***your_element*** > **Notification Rules** and add the configured email delivery endpoint to your notification rule as a subscriber.
   a. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule
   b. Drag and drop the configured contact from the Contacts to the Subscriptions pane.

Update SMTP settings for the email delivery channel

You need to configure the email delivery channel to be able to receive email notifications. This must be done once for each PI AF server.

Procedure

1. To start PI System Explorer, click **Start** > **Programs** > **PI System** > **PI System Explorer**.
2. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
3. Click the **Plug-Ins** tab.
4. Scroll down to the **Delivery Channel Plug-Ins** section, then right-click **Email** and select **Settings**.
5. In the Email Delivery Channel Configuration window, change the following parameters as needed:
   - **SMTP Server**: The fully-qualified domain name of the machine running the SMTP server.
   - **Port**: The listener port of the SMTP Server.
   - **Authentication options**: Options for a primary SMTP server authentication. See Configure authentication for SMTP server connection.
   - **Use TLS**: Option to enable Transport Layer Security (TLS) encryption for the primary server
   - **Backup SMTP Server**: The fully-qualified domain name of the machine running the backup SMTP Server.
   - **Port**: The listener port of the backup SMTP Server.
   - **Authentication options**: Options for a secondary SMTP server authentication.

- **Use TLS**: Option to enable Transport Layer Security (TLS) encryption for the secondary server
- **Sender Email**: The email address from which notification messages will be sent.
- **Allow contacts to set sender email**: Specifies whether individual contacts can change the email address from which their notification emails are sent.
- **Send Timeout**: Time allowed for sent emails to be received by the primary SMTP server before failover to the backup SMTP server occurs.
- **Backup Fail Back Time**: During failover, specifies how long the backup SMTP server sends email before attempting to fail back to the primary server.

Configure a dynamic email delivery endpoint

In PI AF 2017 R2, you can configure an email delivery endpoint as a value of an attribute. This provides an additional flexibility in situations where notification emails can be sent to different recipients without making changes to the notification rule template. To configure a dynamic email delivery endpoint and add an email subscriber to your notification rule, follow this procedure. For a conventional email delivery endpoint configuration, see Configure an email delivery endpoint.

Procedure

1. Navigate to the element that you want to receive the notifications from and add an attribute with your email address as a value. The value of this attribute must be a string. You can enter multiple email addresses delimited by comma.
2. Navigate to **Notification Rules** tab to configure and add an email delivery endpoint as a subscriber to your notification rule.
   a. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.
   b. Click **Create a new dynamic endpoint** under **Dynamic Endpoints**.
   c. Select an email attribute configured in step 1.
   d. Click **Create**.
   e. Drag and drop the configured contact from the Contacts to the Subscriptions pane.
   f. Click **OK**.
3. Go to **Contacts** plug-in to edit or delete a configured delivery endpoint. It will be found in **Delivery Endpoints** folder.

   Note:
   - The email attribute endpoint can be edited from the Subscriptions pane as well.
   - **Test** button in the **Contacts** plug-in is not supported for an attribute-based endpoint at this time.

Configure a REST web service delivery endpoint

To configure a web service delivery endpoint and add a REST web service subscriber to your notification rule, follow this procedure. For SOAP web service, see Configure a SOAP web service delivery endpoint.

Procedure

1. Add a web service delivery endpoint using the PI System Explorer Contacts plug-in.
   a. Go to the **Contacts** plug-in of the PI System Explorer.
   b. Right-click the **Delivery Enpoints** folder.
   c. Select **New Delivery Enpoint**.
   d. Optional. **Retry interval**. Select the time interval at which PI Notifications Service will contact the web service. The web service will be contacted as frequently as the system allows if zero is entered.
   e. Optional. **Maximum Retries**. Select the number of times PI Notifications Service will try to connect to the web service. Entering zero means there will be no retries.
   f. Select **WebService** from the drop-down list next to **Delivery channel**.
   g. Select **REST**.
   h. Enter the URL of your web service.
   i. Select **Http Method**, a default web method to be used for the notification. Supported methods are POST, PUT and PATCH.
   j. Select **Authentication Option**.
      - Windows: the default option. PI Notifications Service will pass the network credentials of its service account to the web service
      - Anonymous: no credentials will be provided to the web service
      - Basic: enter a username and password the service will provide to the web service
   k. Check in the changes.

   See **Create a delivery endpoint such as a stand-alone email or a web service** row in Manage contacts and Options for the notifications web service delivery channel for more information. See also Configure authentication for a web service connection for authentication of web service.
2. Add the configured web service delivery endpoint to your notification rule as a subscriber.
   a. Go to the **Notification Rules** tab of the element you want to add the web service as the delivery endpoint.
   b. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.
   c. From the Contacts pane on the right-hand side, expand **Delivery Endpoints** and drag and drop the configured web service to the Subscriptions pane on the left.
   d. Configure the REST web service.

      REST can send a JSON object to the REST server in HTTP methods PUT, POST or PATCH. Click the "wrench" icon to configure the JSON object for the REST API method.

      Currently, only JSON payloads are supported; query string parameters are not supported.

      If you hover over the "Help" icon , you see sample JSON parameters (path and value) that are necessary for configuring this method:

Enter the path and value on each row. You can validate the method using the **Test Send** button.

Configure a SOAP web service delivery endpoint

To configure a web service delivery endpoint and add a SOAP web service subscriber to your notification rule, follow this procedure.

Procedure

1. Add a web service delivery endpoint using the PI System Explorer Contacts plug-in.
   a. Go to the **Contacts** plug-in of the PI System Explorer.
   b. Right-click the **Delivery Enpoints** folder.
   c. Select **New Delivery Enpoint**.
   d. Optional. **Retry interval**. Select the time interval at which PI Notifications Service will contact the web service. The web service will be contacted as frequently as the system allows if zero is entered.
   e. Optional. **Maximum Retries**. Select the number of times PI Notifications Service will try to connect to the web service. Entering zero means there will be no retries.
   f. Select **WebService** as the **Delivery channel**.
   g. Select **SOAP**.
   h. Enter the web service address, the URL of your web service. You can validate the connection using the **Get Web Services** button.
   i. Enter the name of the web service to be used for notification in the **Web Service** field.
   j. **Default Web Method**: select the default web method to be used for the notification. This menu displays all of the parameters defined in the web service.

k. Select **Authentication Option**.

- Windows: the default option. PI Notifications Service will pass the network credentials of its service account to the web service
- Anonymous: no credentials will be provided to the web service
- Basic: enter a username and password the service will provide to the web service

l. Check in the changes.

See **Create a delivery endpoint such as a stand-alone email or a web service** row in Manage contacts and Options for the notifications web service delivery channel for more information. See also Configure authentication for a web service connection for authentication of web service.

2. Add the configured web service delivery endpoint to your notification rule as a subscriber.

a. Go to the **Notification Rules** tab of the element you want to add the web service as the delivery endpoint.

b. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.

c. From the Contacts pane on the right-hand side, expand **Delivery Endpoints** and drag and drop the configured web service to the Subscriptions pane on the left.

d. Configure the SOAP web service.

SOAP can call the web service methods. Click the "wrench" icon to configure the SOAP API method. The "Information" icon gives you information about the parameters (path and value) that must be configured for this method, and their complexity. Enter the path and value on each row (the path depends on how the web service server is configured); clicking on a row brings up a drop-down list of expected values for that path. The interface also automatically validates the parameters that you enter on each row. You can validate the method using the **Test Send** button.

You have the option to cancel or retry a connection that is taking a long time to complete; this may happen if you have a high latency (commonly called "laggy") connection where the WSDL information is not quickly retrieved.

> **Note:**
> Complex data may be required by the method. To enter the correct information, refer to the Web Service Definition Language (WSDL) documentation for the API method.

Contacts

In PI System Explorer, the Contacts plug-in provides you with contact options for individual users and groups.

Contacts plug-in versions

PI System Explorer works with different versions of the Contacts plug-in.

- To work with the version that is compatible with the new version of notifications that was released in 2016 R2, execute the 64-bit version of PI System Explorer.
- To work with the legacy version that is compatible with legacy notifications, execute the 32-bit version of PI System Explorer. Note that if legacy notifications is not installed, the new version of the Contacts plug-in will load. For more information on PI System Explorer versions, see PI System Explorer.

Subscriber configuration

You can configure subscribers from these types of contacts:

- **Delivery endpoints**

  A delivery endpoint is an entity to which notifications can be sent, and is typically an individual user or an application.

- **Groups**

  A group contact is an unordered collection of delivery endpoints, other groups or escalation teams. Notification messages are sent to all members of the group simultaneously.

  > **Note:**
  > Group contacts that you create in PI System Explorer are not the same as Active Directory (AD) groups.

- **Escalation teams**

  An escalation team is an ordered collection of delivery endpoints or groups.

  Notification messages are sent immediately to the first contact on the list. If the event frame matching the notification rule is not acknowledged or closed within a specified time, notification messages are sent sequentially to the remaining members of the escalation team until the event frame is acknowledged or closed.

  The escalation period defines the amount of time to elapse before a notification is sent to the next contact on the list.

Search for contacts

Click the icon to open the Search Contacts window in the Contacts pane on the top left of the screen. Currently available search criteria are: Name, Description, Department and Email.

> **Note:**
> You must be running the 2018 or later version of PI AF Server to use email as search criterion. Note that search by email address only works for contacts in Active Directory but does not find contacts created manually.

Topics in this section

- Configure Active Directory access for contacts
- Manage contacts
- Options for the notifications email delivery channel
- Options for the notifications web service delivery channel

Configure Active Directory access for contacts

When you use notifications with PI AF server, you may need to specify how to access Microsoft's Active Directory to retrieve contact names for the PI Notifications Service Contacts lists.

Each PI AF server provides the option to specify the domain and contact sub-folder, as well as the account needed to access Active Directory and retrieve contact names. By default, the account under which the PI AF server application service is running is used for Active Directory access. To use a different account or to access an Active Directory in a different domain, configure access from the Configure Active Directory Access for Contacts window.

Note:

- Beginning with PI AF 2017 R2, Active Directory group is shown by default under **Contacts** in PI System Explorer once it is configured.
- Notifications 2016 R2 or later will automatically handle a change in email address of the user or a group in Active Directory by contacting the domain controller before sending an email each time.

Procedure

1. Open PI System Explorer and connect to a database on the PI AF server for which you want to configure Active Directory access.
2. Click **File** > **Server Properties**.
3. In the PI AF Server Properties window, click the **Configure Active Directory Access for Contacts** link.
4. In the **Active Directory Domain Name** text box, enter the full DNS name of the Active Directory domain from which the contact names will be retrieved for the PI Notifications Service Contacts (for example, `contoso.com`).

   If this field is left blank, the domain in which the PI AF application service resides will be used.
5. In the **Active Directory Contact Sub-Folder** text box, enter the path to the folder containing the list of contacts for this domain.

   In larger Active Directory domains, contacts may be organized within sub-folders. The use of sub-folders can allow for faster retrieval of a list of Active Directory contacts.

   Use the following structure for the sub-folder:
   ```
   DomainUserFolder/SubDomainUserFolder/Sub-SubDomainUserFolder
   ```
6. Choose an option for **Active Directory Access Account**:
   - **Use the account the PI AF Server runs as**

     This is the default option. Select it to access Active Directory using the account under which the PI AF application service runs. By default, the PI AF server is installed using a virtual account, `NT SERVICE\AFService`. However, the PI AF server service account can be changed. If the PI AF server service account does not have the necessary permission to read the Active Directory, no contact names will be retrieved in the Contacts list. If your Active Directory security is configured to allow the PI AF server service account to read the Active Directory, this is the simplest option.
   - **Use the account the AF Client is running as**

Select this option to use the credentials of the user account under which the connecting client application is running. If the PI AF server service is running under an account (a virtual account, `NT SERVICE\AFService` is the default account) that does not have permission to read the Active Directory, this option can be used. As long as the user account under which the connecting client application is running has permission to read Active Directory, a list of contact names is returned to the Contacts list. The contents of the Contacts list may vary, depending upon the access account used, since the security to read the contact list is determined by Active Directory.

> **Note:**
> Specifying this option may require Kerberos configuration if a PI AF SDK application will be using impersonation in a middle tier, such as a Web Service.

- **Use the specified account**

  This option allows you to specify an account to use to read the Active Directory. This can be useful when the Active Directory and PI AF server are in different domains or when the accounts in the first two options have no permission to read the Active Directory. For **Account Name**, use the format `Domain\User`. Make sure the specified account has the appropriate permission to read the target Active Directory.

7. Check **Use Active Directory's locally cached Global Catalog** to use the global catalog for Active Directory domain controller searches. Otherwise searches must go to the owning domain controller.

   Active Directory holds information in a distributed data repository called a global catalog. For installations where there are multiple, distributed domain controllers, each domain controller has a cache of the portions of the global catalog for which it is not responsible, so that Active Directory searches do not have to be referred to the owning domain controller. This improves performance for queries that must otherwise have to access a remote domain controller.

8. Choose a setting for **Return All Persons**.

   Active Directory objects are derived from one another as follows:

   ```
   Top>Persons>OrganizationalPerson>Contact
   ```

   and

   ```
   Top>Persons>OrganizationalPerson>User
   ```

   - Select this check box to return Persons, Organizational Persons, Contacts and Users from the target Active Directory.
   - Clear the check box to return only Users.

Manage contacts

Use the Contacts plug-in to manage individual contact information, as well as manage groups, escalation teams, and delivery endpoints for use with notifications.

Before you start

You cannot nest escalation teams.

Procedure

1. In the navigator, click **Contacts**.
2. In the Contacts browser, choose from the following actions.

To ...	Do this ...
Edit an existing contact	a. Choose one of the following actions: ▪ Enter search criteria in **Search for contacts** and press Enter. Note that * returns all contacts. ▪ Click next to **Search for contacts**, enter search criteria in the Search Contacts window, and click **OK**. ▪ Expand the **Contacts** folder, click **New search...**, enter search criteria in the Search Contacts window, and click **OK**. Search results are displayed in the Contacts browser. b. Click beside the contact you want to update and click the email delivery endpoint. c. In the viewer, you can make the following changes: ▪ In the **Description** field, add additional contact information. ▪ Enter notification contact options in the **Retry interval** and **Maximum Retries** fields.
Create a custom contact that is not already in Active Directory	a. Right-click the **Contacts** folder and click **New Contact**. b. In the viewer, enter a name and an email address. All other information is optional. The contact is displayed in the Contacts browser. c. Click beside the newly created contact. **Note:** The icon indicates non-AD contacts (as opposed to). d. Click the email delivery endpoint and, in the viewer, enter notification contact options in the **Retry interval** and **Maximum Retries** fields.

Create a group	a. Right-click the **Groups** folder and click **New Group**. b. In the **Name** field, enter a unique name for the group. c. Choose from the following actions: ▪ Click and, in the Select a Contact window, locate a contact with a new search, or select another group, an escalation team, or a delivery endpoint, and click **OK**. Repeat as needed. ▪ In the Contacts palette, locate a contact with a new search, or select another group, an escalation team, or a delivery endpoint, and drag it onto the viewer. Repeat as needed. d. Set notification contact options for the group. ▪ Right-click each subscriber delivery endpoint, click **Options**, and enter notification contact options in the **Retry interval** and **Maximum Retries** fields. ▪ If escalation teams are included in a group, right-click each escalation team, click **Options**, and enter notification contact options in the **Escalation period** and select an **If not acknowledged** option.

Create an escalation team	a. Right-click the **Escalation Teams** folder and click **New Escalation Team**. b. In the **Name** field, enter a unique name for the escalation team. c. In **Escalation period**, specify how long you want an escalation to last. d. In **If not acknowledged**, specify the action to be taken if the notification is not acknowledged after being sent to all contacts in a team: ▪ To stop the escalation process, select **End escalation**. ▪ To repeat the escalation process for a specified number of times until the notification closes or is acknowledged, select **Repeat *N* times**. ▪ To repeat the escalation process indefinitely until the notification closes or is acknowledged, select **Repeat while active**. e. Choose from the following actions: ▪ Click and, in the Select a Contact window, locate a contact with a new search, or select another group, or a delivery endpoint, and click **OK**. ▪ In the Contacts palette, locate a contact with a new search, or select another group, or a delivery endpoint, and drag it onto the viewer. f. Configure the sequence for the escalation chain. ▪ Click a subscriber and click and to position as needed. ▪ To remove a subscriber, click the subscriber to be removed and click .

Create a delivery endpoint such as a stand-alone email or a web service	a. Right-click the **Delivery Endpoints** folder and click **New Delivery Endpoint**. b. In the **Name** field, enter a unique name for the delivery endpoint. c. Enter notification contact options for the delivery endpoint in the **Retry interval** and **Maximum Retries** fields. d. From **Delivery channel**, select a deliver option. ▪ For **Email**, enter address configuration information, as described in Options for the notifications email delivery channel. ▪ For **WebService**, enter web service address configuration information, as described in Options for the notifications web service delivery channel.

3. To save changes you have made to contacts, press Ctrl+S or click **Check In**.

Options for the notifications email delivery channel

An email delivery endpoint can be configured in ways specific to the email delivery channel, such as the address to which notification is sent. The following table shows options that are available for all email delivery endpoints:

Option	Description
`To Email`	Email address to which the notification is sent. This option cannot be changed in the default email delivery endpoint that was derived from the Active Directory (AD) contact. Multiple email addresses can be entered delimited by comma.
`From Email`	Email address from which the notification is sent. This option is available solely if your PI System manager has configured the email delivery channel with the setting `Allow contacts to set sender email`.
`Use HTML formatting`	Specifies if the delivery endpoint receives messages in HTML format. HTML formatting makes messages easier to read and allows messages to contain hyperlinks.

Options for the notifications web service delivery channel

Creating a web service delivery endpoint

Using the Contacts plug-in of the PI System Explorer (PSE), you can configure a web service delivery endpoint for SOAP or REST web services. To map parameters for the API call, create and configure body of the message, see Configure a SOAP or REST web service delivery

endpoint. For authentication of a web service, see Configure authentication for a web service connection.

Configuring the SOAP web service

The following table lists the options you must set for a SOAP web service delivery endpoint:

SOAP web service options

Option	Description
`Web Service Address`	The URL of your web service. You can validate the connection using the **Get Web Services** button.
`Web Service`	The name of the web service to be used for notification.
`Default Web Method`	Default web method to be used for the notification. This menu displays all of the parameters defined in the web service.

The PI System Explorer user interface automatically retrieves the associated Web Service Definition Language (WSDL) methods, provides useful information about the parameters, and guides you through configuring the web service.

Configuring the REST web service

The following table lists the options you must set for a REST web service delivery endpoint:

REST web service options

Option	Description
`URL`	The URL of your web service.
`HTTP Method`	Default web method to be used for the notification. Supported methods are POST, PUT AND PATCH.

Configuration of notification rules for analyses or event frames

The asset analytics and notifications features of the PI AF server together provide effective ways for you to perform condition-based or predictive maintenance. Notification rules leverage the event frame feature of PI AF to notify you of important events that affect your system.

This section guides you in creating notification rules for your analyses or event frames.

Topics in this section

- Configure notification rules from analyses
- Configure notification rules for user-defined event frames
- Configure escalations for notification rules

Configure notification rules from analyses

If you have already configured analyses to detect anomalies in your PI AF asset data, or you wish to configure such analyses, follow this procedure to configure notification rules on configured analyses.

Before you start

For information on software and system requirements, see Requirements for notifications.

> **Note:**
> If you want to configure notification rules to trigger on event frames from sources like event frame interfaces or custom applications, see Configure notification rules for user-defined event frames.

For information on creating an event frame generation analysis, see Event frame generation analyses.

Procedure

1. Select the existing event frame generation analysis for which you want to create a notification rule.

2. Click **Create a new notification rule for the selected analysis**. You see the **Notifications Rules** tab highlighted, as well as the options to configure a notification rule with the default name "Notification Rule".

3. Edit the created notification rule.

   Provide one or more of the specified criteria; some of your criteria may be pre-selected from your analysis. The notification rule is triggered when all criteria are satisfied. You may

change the name of the notification rule from the default, add a suitable description, and select a suitable category.

Selecting a category aids in organizing and searching for your notification rules.

4. Click **View/Edit Trigger**

   The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**. Make sure that the **Analysis** criteria mode is selected.

5. Configure options in the Trigger Criteria pane.

   - You see the details of the selected analysis such as the event frame template name, the start and end triggers, and so on.
   - You can configure a new analysis of type **Event Frame Generation** or **SQC**. When you choose to create a new analysis with either the default or a custom name, you see the same analysis creation window as from the **Analyses** tab of PI System Explorer.
   - For more information, see Trigger criteria in notification rules.

6. Configure options in the Options pane.

   - Optional. **Resend Interval**. Select the time interval at which PI Notifications Service will send additional alerts if the event frame matching the notification rule is not acknowledged or closed. For more information, see Resend Interval in notification rules.
   - Optional. **Non-repetition Interval**. Select the time interval for which PI Notifications Service will not send similar alerts associated with the same notification rule. For more information, see Non-Repetition Interval in notification rules.
   - **Event Frame Can Be Acknowledged**. This option is automatically selected if the event frame template has been configured for acknowledgement.
   - Optional. Options based on multiple start triggers. If you have configured multiple start triggers for your analysis, choose between the options of being notified (a) when the current trigger severity is higher than any trigger severity encountered so far, (b) when the current trigger severity is higher than the previous trigger severity, or (c) every time a trigger condition is true, regardless of its relative severity to other previous triggers.

7. Optional. To configure the email format, click **Manage Formats** in the Subscriptions pane. You see the Message and Content panes. By default, the notifications feature uses the Global

Default Email format for each subscriber. For more information, see Delivery formats in notification rules.

> **Note:**
> If the Content pane has a message indicating that your selected format is "read-only", you must first add a new format before you can customize it.

8. Optional. To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon.

   The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names. To add more information to your custom format, drag and drop information from the Content pane to the Message pane.

   > **Tip:**
   > A hyperlink indicates that an event frame template is selected; you may add the attributes defined below the event frame template to the content of your email notification by dragging and dropping the attributes from the Content pane to the Message pane.

   You can designate your custom format as the default by clicking the **Is Default** box in the Message pane.

9. Click **View/edit subscriptions** in the Subscriptions pane.

   You can search for contacts using **Contacts Search** in the Contacts pane. To add a subscriber, drag and drop a specific contact from the Contacts pane to the Subscriptions pane. Make sure to select your customized email format if you have created one, or pick the Global Default Email format. For more information, see Subscriptions in notification rules.

10. Optional. Configure escalation teams.

    You can configure members of escalation teams from the Contacts plug-in on PI System Explorer, as described in Manage contacts. The escalation team contacts or groups will be notified one by one at the end of time segments specified in **Escalation period**. You can also control the frequency of the escalation notification.

    > **Note:**
    > Escalations are triggered on event frame templates that can be acknowledged. Open the **Library** plug-in on PI System Explorer, select your event frame template and verify that the **Event Frame Can Be Acknowledged** check box is selected.

11. Check in the notification rule.

    When the event of interest occurs, an email will be sent to the subscriber's mailbox. The email shows the event frame that triggered the notification rule with customized details that you have configured.

After you finish

To manage your notification rules, use the **Management** plug-in on PI System Explorer. See Management of notification rules.

Configure notification rules for user-defined event frames

Use this procedure if you want to configure notification rules to trigger on event frames from sources like event frame interfaces or custom applications.

Before you start

For information on software and system requirements, see Requirements for notifications.

> **Note:**
> If you have already configured an analysis on your PI AF asset, or wish to create new analyses to generate the event frames, use the procedure Configure notification rules from analyses.

Procedure

1. In the PI System Explorer navigator, click **Elements** and select the element for which you want to define the notification rule.
2. Select the **Notification Rules** tab.
3. Click **Create a new notification rule**.

Change the name of the notification rule from the default, add a suitable description, and select a suitable category. Selecting a category aids in organizing and searching for your notification rules.

4. In the Trigger pane, click on the hyperlink **Please configure trigger criteria for this Notification Rule**.

   The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**.

5. Select the **Event Frame Search** trigger criteria mode.

6. Configure options in the Trigger Criteria pane.

   You can select the event frame template from the drop-down list, and add conditions using any of the attribute templates for the event frame template.

7. Configure options in the Options pane.

   - Optional. **Resend Interval**. Select the time interval at which PI Notifications Service will send additional alerts if the event frame matching the notification rule is not acknowledged or closed.
   - Optional. **Non-repetition Interval**. Select the time interval for which PI Notifications Service will not send similar alerts associated with the same notification rule.

   Note:
   Verify that the event frame template has the **Event Frame Can Be Acknowledged** check box selected.

8. Optional. To configure the email format, click **Manage format** in the Subscriptions pane.

   You see the Message and Content panes. By default, notifications uses the Global Default email format for each subscriber.

   Note:
   If the Content pane has a message indicating that your selected format for a subscriber is "read-only", you must first add a new format before you can customize it.

9. Optional. To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon.

   The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names. To add more information to your custom format, drag and drop information from the Content pane to the Message pane.

   > **Tip:**
   > A hyperlink indicates that an event frame template is selected; you may add the attributes defined below the event frame template to the content of your email notification by dragging and dropping the attributes from the Content pane to the Message pane.

   You can designate your custom format as the default by clicking the **Is Default** box in the Message pane.

10. Click **View/edit subscription** in the Subscriptions pane.

    You can search for contacts using **Contacts Search** in the Contacts pane. To add a subscriber, drag and drop a specific contact from the Contacts pane to the Subscriptions pane. Make sure to select your customized email format if you have created one, or pick the Global Default email format.

11. Optional. Configure escalation teams.

    You can configure members of escalation teams from the Contacts plug-in on PI System Explorer, as described in Manage contacts. The escalation team contacts or groups will be notified one by one at the end of time segments specified in **Escalation period**. You can also control the frequency of the escalation notification.

12. Check in the notification rule.

    When the event of interest is generated, an email is sent to the subscriber's mailbox. The email shows the event frame that triggered the notification rule with the customized details that you have configured.

After you finish

To manage your notification rules, use the **Management** plug-in on PI System Explorer. See Management of notification rules.

Configure escalations for notification rules

An escalation team is an ordered collection of delivery endpoints or groups. Notification messages are sent sequentially to the members of escalation teams until the event frame matching the notification rule is acknowledged or closed. You can set up an escalation structure to ensure that the right people are notified with the right information, and in the right order.

Subscribe an escalation team to send notification messages sequentially to a group of contacts. For more information on creating an escalation team, see **Create an escalation team** in Manage contacts.

Management of notification rules

To manage your notification rules, use the Management plug-in, accessible on the bottom left of the main screen of the PI System Explorer. When accessed, the plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Notification Rules** button to view and manage your notification rules. For more information, go to Management of notification rules. To search for notification rules configured in your database, see Search for analyses or notification rules.

Changes from PI Notifications 2012

The notifications feature is based on event frames and does not have an analysis engine like PI Notifications 2012 and earlier versions (henceforth referred to as "legacy" versions). Notification rules leverage PI Analysis Service, as well as other applications that generate event frames, to identify important events in the PI Server, to use event frames as the container for relevant information for the event, and to use annotations on the event frame to store notification history.

The notifications feature is driven by event frames; therefore, some legacy notification features are either different or rendered obsolete. A migration tool is included to enable migration of notifications from your PI Notifications 2012 legacy version to notification rules, thereby preserving the investment you made in your legacy system.

> **Note:**
> If you choose to migrate, the migration tool converts your legacy notifications into analyses and notification rules.

For more details on the migration tool, see Migration from PI Notifications 2012.

User interface changes from PI Notifications 2012

Notifications in PI Server 2016 R2 (and later versions) is fundamentally different from the legacy PI Notifications 2012, and you cannot share configurations between the two systems.

To configure event-based notifications, you must use the **Notification Rules** tab in the **Elements** view of 32-bit or 64-bit PI System Explorer. To manage legacy notifications, you must use the **Notifications** plug-in on the 32-bit PI System Explorer.

Migration from PI Notifications 2012

PI Server 2016 R2 and later versions include a migration tool to enable migration of your 2012 notifications to notification rules based on event frames. The migration tool is run at the end of the PI Notifications Service install, and you may choose to migrate your legacy notifications at that time.

You may also run the migration again from PI System Explorer, after the installation is complete. To run the migration tool from PI System Explorer, make sure that all these conditions are true:

- PI Notifications Service is installed on your machine
- The migration tool executable file `PINotificationsMigrationTool.exe` is installed on your machine, typically in the `PIPC\Notifications` folder

The migration tool causes these object migrations:

- Legacy notification triggers are migrated as event frame generation analyses and are run in PI Analysis Service.
- Legacy notifications created in previous versions are migrated to notification rules, and they run in the PI Notifications Service.

- Legacy notifications history is fully migrated and created as event frames. History in PI points are migrated as annotations on event frames.
- Legacy notification template name, description and properties are mapped to the notification and target properties; a message is also logged in the migration report.

> **Note:**
>
> Properties of the notification rule template cannot be overridden on a notification rule based on the template with an exception of the description field.
>
> Due to this, if a legacy notification deviates from the legacy notification template, that information may be lost during migration. For example, if the legacy notification template is named "Notification for Pumps", but a legacy notification based on this template is named "Notification for Pumps - Houston", the migrated notification rule template and all notification rules based on this template will be named "Notification for Pumps". The description field of the migrated notification rule, however, will contain the name of the legacy notification. The migration report will contain the name changes and any other changes that have been made.

Training video

For more information on migration, watch the OSIsoft Learning channel video:

https://www.youtube.com/watch?v=bGdN6VrBVUw&rel=0

Running the migration tool

Before you begin migration you must verify that you have `read` permissions on the PI Data Archive where the legacy notifications history is stored and `read/write` permissions on the PI AF database where the legacy notifications are configured.

You may need to back up the PIFD (SQL database) if you want to re-run the migration and avoid having duplicate notification rule and analysis objects created. If you re-run your migration and are warned that duplicates may be created during the re-migration, you can first restore your PIFD database from the backup and avoid the creation of duplicates. For instructions on how to perform a backup of the PIFD database, see the PI Server topic "OSIsoft Backup job" in Live Library (https://livelibrary.osisoft.com).

You can re-run the migration tool by clicking **Tools** > **Notifications Migration** from PI System Explorer. The migration tool first determines and displays information on your state, which is one of:

- **Not migrated**

  Legacy notifications may be created and managed using the **Notifications** plug-in on the bottom left of the PI System Explorer window. You must use the 32-bit version of PI System Explorer to manage legacy notifications.

  You can choose to be in the "Not migrated" state by choosing the appropriate option while running the migration tool. In this state, you cannot create new notification rules.

- **Migrating**

  Legacy notifications are read-only; only new notification rules may be created. You can stop and start legacy notifications while in this state.

In the migrating state, you may run and test legacy notifications and new notification rules side-by-side to ensure that the migration is successful and that the new analyses and notification rules are working as intended. You can check the migration report, fix the errors indicated on legacy notifications, and then choose to re-run migration for only those legacy notifications that encountered errors in the previous migration run.

You may move to the migrated state when you have successfully tested that your migrated notification rules match your legacy notifications. Move to the migrated state only when the following are true:

- You intend to continue to work only with the new notification rules
- You will never need to revert to your legacy notifications
- You have successfully migrated all the legacy notifications that you wanted to migrate

- **Migrated**

  When you move to this state, you can create only new notification rules in the system. Your legacy notifications are deleted and you cannot revert to your legacy notifications or create legacy notifications.

  From the migrated state, you may re-run the migration tool to revert to the "Not migrated" state; however, your legacy notifications cannot be recovered.

Sample migration report

The migration tool generates a report when it completes the migration process. The report includes information on whether legacy notifications are migrated successfully. The "Errors" section and the "Not Supported" sections indicate notifications that are not migrated. The "Warnings" section indicates notifications with changed functionality, and the report provides more information.

> **Note:**
> Migration report can be found in `%userprofile%\Documents` folder of the person running the migration.

This is a sample migration report summary:

Notifications Migration Report for TMIPVDSHD01

Summary

Started: 8/24/2016 1:40:02 PM
Ended: 8/24/2016 1:59:07 PM
Notification templates: 26 of 32 migrated successfully
Notifications: 1736 of 1832 migrated successfully
Notifications or templates with errors: 92
Notifications or templates with warnings: 10
Notification Rules created: 4503
Analyses created: 4503
Legacy history instances processed: 0
Event frames created: 0
Warnings (2)
1. Acknowledgment Web Page hyperlink has been migrated to a PI Coresight event details hyperlink.
2. PI Coresight event details hyperlink requires a PI Coresight installation.

Notification Templates (32)

Outcome	Element Template	Legacy Notification Template	Migrated Notification Rule Template	Migrated Analysis Template	Messages
Not Migrated (5)					
There was an error migrating legacy notification templates. The notification templates below have not been migrated, please see the messages listed below for more information on how to resolve the issue.					
		P442_LIGNE_PANNE_COMM			Errors (1) 1. The notification had no target element.
		P442_TRANSFO_PANNE_COMM			Errors (1) 1. The notification had no target element.
		UM7650_PANNE_COMM			Errors (1) 1. The notification had no target element.
	\AADT_DEFAUT\ElementTemplates[SA_BAT]	SA_PTROSEE			Errors (1) 1. Monthy or weekly periodic time rule is not supported.
	\AADT_SUPERVISION\ElementTemplates[Interface PICybectecSMP-PROD/Frontal]	Etat_Interface_PI-Cybectec_SMP_ou_Frontal DPTNE			Errors (1) 1. The analysis is not convertible.
Migrated With Altered Configuration (1)					
The legacy notification templates were migrated to analysis templates and notification rule templates. During migration some changes were made to the migrated analysis templates and notification rule templates. No changes were made to the legacy notification templates. Please see the messages below for more information on the changes that were made.					

In addition to the summary, the report also contains information on legacy and migrated notification rules and templates, and specific information such as notification rules and analysis names, number of migrated event frames, and the number of processed history events.

Additional resources

Training video

For more information on using notifications in PI Asset Framework, watch the videos on the OSIsoft Learning channel playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2Jbcs43wQ2qO2dxW72PqGhZjKW&rel=0

Management of analyses and notification rules

To manage your analyses or notification rules, use the Management plug-in, accessible on the bottom left of the main screen of the PI System Explorer. The Management feature (which includes analyses and notification rules) is available if your installation of PI System Explorer includes the Management plug-in. You can view and manage the analyses or notification rules in the current PI AF database. In the **Management** pane, search for analyses or notifications, backfill or recalculate your analyses and enable or disable analyses and notification rules in bulk.

Topics in this section

- Search for analyses or notification rules
- Management of analyses
- Management of notification rules

Search for analyses or notification rules

Procedure

1. Navigate to the Management pane.
2. Select what you want to search: **Analyses** or **Notification Rules**.
3. Use the provided search or create your own.

To	Then
Use a provided search	Select **All**, **Enabled** or **Disabled**. These are view-only and cannot be modified, as indicated by the (**View selected search**) icon.

Create a custom search	**Note:** Customized search is only visible to the user who created it on the computer where it was created. Creating your own search is a new feature in PI AF 2017. Refer to the previous versions of PI System Explorer User Guide if you are using an earlier version of PI AF. a. Click the (**Add new search**) button. b. Select a search criteria. Choices are: ▪ Name ▪ Description ▪ Element Name ▪ Template ▪ Category ▪ Enabled/Disabled: enabled or disabled ▪ Service Status: PI Analysis/Notifications Service status ◦ Error ◦ Running ◦ Stopped ◦ Suspended ◦ Warning **Note:** **Service Status** is based on the PI Analysis Service connection and its running status. This cannot be combined with other search criteria.
View/edit a search	Click the (**View/edit selected search**) button to view or edit your search.
Delete a search	To delete, click the (**Delete selected search**) button.

Management of analyses

The **Management** plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Analyses** button to view and manage your analyses.

Note:
For information on notifications management, see Management of notification rules.

Analyses pane

The **Analyses** pane displays information about selected analyses in these columns:

- Status
- (Automatic recalculation)

- (Analysis Type)
- Element
- Name (of analysis)
- Template
- Backfilling (status)

> **Note:**
> In the **Analyses** pane, you may see the number of analyses depicted with the "~" character, or with the word "about"; for example: "~1000" or "1-500 of about 1000". In some cases, the exact number of analyses is not known until they are all loaded.

- **Analysis selection**

  With PI Server 2015 R2 and later versions, analyses results are displayed in pages of 500 results each. You can select:

  - Specific analyses by clicking on each individual analysis check box.
  - All 500 results on a page by clicking the checkbox on the top-left corner.
  - (When more than 500 analyses exist.) All analyses on all pages by clicking **Select all analyses on all pages**.

    > **Note:**
    > Clearing specific analyses automatically makes your subsequent operation page-based. For example, if you first select all 1000 analyses, and then clear two, the operation will be performed on 498 (500 minus 2) and not 998 (1000-2) analyses.

- **Operations**

  You can enable, disable, backfill/recalculate the analyses you have selected. For specific selection criteria, see the **Analysis selection** section above. From PI AF 2017 R2 and later, you can enable or disable automatic recalculation of the analyses you selected. For more information, see Enable or disable automatic recalculation for multiple analyses.

  In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. Note that manual recalculation is only available for PI Analysis Service 2016 R2 or later. In addition, PI Data Archive that stores PI points must be running version 2016 R2 or later. For more information, see Backfill or recalculate data for multiple analyses.

  Expression, rollup, or SQC analyses can backfill or recalculate data if the analysis outputs are mapped to PI point attributes. If PI points already have data for the backfill time period, you can retain existing data and fill in missing data, or delete the data and recalculate.

  Event frame generation analyses can also recalculate event frames over a specified time period. The recalculation process, regardless of the selected option, automatically deletes all existing event frames for that time period, as well as annotations on affected event frames.

- **Pending Operations**

  See the status of following bulk operations: enable, disable, backfill or recalculate a group of analyses.

- **Analysis details**

  Select an analysis to view detailed information.

  - To see a summary of the analysis configuration and status, click the **Overview** tab.
  - To see error details, click the **Errors and Warnings** tab.
  - To view or edit the selected analysis, click **Analysis Configuration** to go to the **Analyses** tab of its associated element.

Management of notification rules

The **Management** plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Notification Rules** button to view and manage your notification rules.

Notification Rules pane

The **Notification Rules** pane displays information about selected notifications rules in these columns:

- Status
- Element
- Notification rule name
- Template (Notification rule template)
- Categories

You can select or deselect notification rules from the list of configured notification rules, and monitor individual status on the Notification Rule Details pane; you can also enable or disable selected notification rules from the Operations pane.

Operations, Pending Operations, and Notification Rule Details panes

The other screens on the **Management** plug-in include:

- An **Operations** pane where you can enable or disable notification rules.
- A **Pending Operations** pane that provides information on whether an operation is queued or complete.
- A **Notification Rule Details** pane that is displayed when you select a notification rule, and has details, status and error information about the notification rule. Click on **Notification rule configuration** to go back to the element **Notification Rules** tab.

Process models in PI AF

A PI AF model consists of connected elements that represent a logical model of your process. A model is itself an element, but with two additional element properties: layers and connections. Elements in the model can represent physical entities in your process, such as tanks, pipes, and process units, or logical entities, such as recipes and summary data. The model is composed of connected elements.

PI AF models can be simple, containing only a handful of elements, or they can be very complex, containing thousands of elements and measurements. The size of the model is only limited by the data available to fulfill the information requirements of an analysis on the model.

Topics in this section

- The scope of a PI AF model
- Guidelines for PI AF models
- Submodels
- Element types in models
- Create PI AF models
- Edit PI AF models
- Ports and connections
- Layers

The scope of a PI AF model

While the number or variety of elements that a model encompasses does not change how a model is stored, it does help in planning both the initial model design and the information needed to complete a model analysis and get meaningful results. When considering the scope of a model, remember that it needs to be small enough for the relationships between properties to be well defined, but large enough to include some redundant data. Redundant data can be calculated from other data in the model.

At different levels of scale, or modeling scope, the consumers of the resulting information will change. For example, an engineer looking at equipment performance of a section of the facility may need more detailed information than a resource planning individual who tracks materials throughout a facility. The planner needs measurements in the model that are different from the measurement needs of the engineer. Elements that are common to both models use the same source elements; this aids in the construction of a variety of models using the same element library.

Three modeling levels, in order of increasing scope, are the following:

- Unit Model
  - This level of modeling typically includes the smallest details of information within a processing unit or area. This scope is useful for monitoring equipment performance and is used primarily by the engineers on that equipment. Auxiliary loops and heating

systems that occur at this modeling level might not have influence outside of this area, and therefore would not be included at the next scale.

  - To perform a meaningful analysis on a unit level model, detailed data must be available on the materials and quantities within the unit. If there is only information available on the parameter of the unit (the inputs and the outputs), this is an indication that the data model is at its smallest granularity.

- Multi-Unit Model
  - When considering unit-to-unit type models, detail within the unit is typically summarized by the connections to and from that unit. Material added and sent to storage areas (tanks, stockpiles, etc.) would also be included.
  - At the multi-unit modeling level, measurements within the unit that do not affect the main inputs and output quantities are not included in the model.
- Boundary Model (Facility and Business Unit)
  - It is useful to create an analysis for the materials entering and leaving the "fence line" of the facility, further summarizing the unit-to-unit information to only the transactions to and from the process from material transfer points (shipping docks, weigh scales, tank feeds, pipelines). This level is also useful to analyze transactions between business units within a facility.

Guidelines for PI AF models

Consider the following rules as you set up a model for analysis:

- Use a systematic approach to identify the location and calibration setup for each instrument in the model.
- Associate every instrument with the correct flow, and make sure that temperature and specific gravity corrections are applied to flow measurements if needed. Check whether this function of correction is performed by your control system or historian.
- Validate tank/inventory properties by building a tank-only model, including shipments and receipts. Enter the materials list with its properties at the same time.
- Determine the calibration tolerance of each instrument (you can use ISO 5167 for orifice meters), or refer to meter data sheets and manufacturers.
- Build and examine individual unit models before attempting to connect them and run a multi-unit analysis.
- After you achieve full unit connectivity, examine the entire drawing, and then run a balance analysis to identify flows and connections that cannot be solved.

Submodels

A model is also an element, which means that a model can be composed of other models - referred to as submodels. This allows for either top-down or bottom-up development of a plant model. The boundary elements of a model normally define the elements with ports that can be used for connections outside the model.

Element types in models

Six core element types are used to enforce connectivity rules between elements of a model.

- Use a **Node** to represent a physical entity in your model, such as a tank, valve, or process unit.
- Use **Measurement** to indicate that the element is used for ascertaining dimensions, quantities, capacities, etc., such as, meters or scales.
- Use **Flow** to indicate that the element carries material from one element to another.
- Use **Transfer** as a temporal flow. The existence of a transfer in a model is only available in the context of time, for example, in a case. Transfers can also be accessed by performing time-based searches of the database.
- Use **Boundary** to define the input and output ports for the model.
- Use **Other** to represent a logical collection of attributes, such as a recipe.

In addition to the core element types, two other types are supported: **Any** and **None**. Use the **Any** and **None** element types when you define the connectivity rules for a port.

Create PI AF models

You create a PI AF model in the **Elements** browser.

Procedure

1. In the navigator, click **Elements**.
2. Right-click the collection of elements and click **New Model**.
3. In the Choose Model Template window, select a template on which to base the model or select **None**.
4. Click **OK**. The model configuration tabs appear in the viewer.
5. In the **Name** field of the **General** tab, enter a name for the model.
6. Optional. In the **Description** field, enter a description for the model.
7. The read-only **Template** and **Type** fields list the template and element type chosen when the model was created. To view the template properties, click . Element types are described in Element types in models.
8. Optional. You can organize objects by grouping them into categories. To browse the available categories, click .
9. To assign a default attribute, select the attribute from the **Default Attribute** drop-down list. Note that you must add attributes in the **Attributes** tab first before they appear in the list. This field is read only if the model is based on a template; the box displays the attribute specified in the template.
10. Optional. To specify location attribute traits for the model, click the **Location** link. For more information, see Set location attribute traits.

11. Optional. To specify asset health attribute traits for the model, click the **Health** link. For more information, see Set health attribute traits.
12. Optional. To configure access permissions for the new model that are different from those inherited from the Elements collection, click the **Security** link. For more information, see Configure security for objects.
13. Click **OK** and check in your work.

Edit PI AF models

You view or edit PI AF models in the <Model Name> Connections window.

Procedure

1. Right-click on a model in the **Elements** browser and click **Model Connections**.

   The <Model Name> Connections window shows a visual representation of the elements in the model and how they are connected.

2. To edit the model:
   - Right-click an element in the **Connections** pane to create a new connection or to view or edit the properties of the element.
   - Click an element to center it in the **Connections** pane and show any other connected elements. You can move along the flow of the model this way, element by element.
   - Click a connection to view or edit the connection or to add another source or destination.
   - Right-click a connection to make another connection, view the properties of the selected connection, or to copy and paste the properties of the connection.

Ports and connections

Elements in a model are connected through any number of ports, which are defined by the element template. A port can be defined as an input port, an output port, or as an undirected port. The port defines how many connections can be made and the types of elements that can be connected.

In analyses, directed ports (input and output ports) represent positive material flow and are used by connections. Undirected ports are used by attachments of meters and analyzers. The most common type of attachment in a model is a measurement or meter attached to a flow element.

A connection represents the link between the ports of two elements. The ports, which are defined by the element template, can be defined as input ports, output ports, or undirected ports.

Topics in this section

- Create ports
- Create connections

Create ports

Before you start

To specify a port as the default port, you must open the Properties window.

Procedure

1. In the Elements browser, select an element and then click the **Ports** tab in the viewer.

   > **Note:**
   > If the element is based on a template, you cannot add a port unless the template allows extensions.

2. Click **New Port**, and create a port for the element. Right-click an existing port to view or edit its properties.
3. Configure the port.
   - **Port Type:** Select the port type: **Input**, **Output**, or **Undirected** (for meters, for example).
   - **Allowed Categories:** Select the categories of which the port is allowed to be a member.
   - **Maximum Connections:** Specify the maximum number of connections that can be made to the port. Enter zero for an unlimited number of connections.
   - **Connection Type:** Select the type of element to which the port can connect, for example, Node, Boundary, Measurement, and so on.
   - **Allowed Templates:** Select elements allowed to connect to the port. Select only elements created from the selected template.

Create connections

Use the **Connections** tab to display the connections in a model.

Procedure

1. To create a new model in the Elements browser, right-click the collection of elements as described in Create PI AF models, and click **New Model**.
2. Follow these steps to fill in the **Connections** tab.
   a. Right-click in the field and select **New Connection**.
   b. In the Make Connection window, right-click an existing connection to view or edit its properties.
   c. In the **Source** field, select the source for the connection. Select the appropriate port of the source in the corresponding **Port** field.
   d. In the **Destination** field, select the destination for the connection. Select the appropriate port of the destination in the corresponding **Port** field.

   To include child elements, select the **Include Child Elements** check box.

Layers

You can organize the elements of a model into layers. Layers provide a mechanism for including or excluding portions of the model as needed for analysis. An element of a model can belong to

more than one layer. You can also use the Layers feature with a graphic modeling tool, such as PI ProcessBook, to provide a visual overlay functionality.

Create layers

Create a Layer to enable you to include or exclude portions of a model.

Procedure

1. In the Elements browser, select the model to which you want to add a layer.
2. Click the **Layers** tab in the viewer.
3. Right-click and click **New Layer**.
4. In the **Name** field of the Layer Properties window, enter a unique name for the layer.
5. Optional. In the **Description** field, describe the purpose of the layer.
6. In the element list, select one or more elements that comprise the layer.
7. Click **OK**.

PI AF utilities and plug-ins

After you install PI AF, several utilities are available to assist you in your administration of PI AF and management of plug-ins.

Topics in this section

- Launch PSE with command line options
- PI AF Diagnostics utility
- AF Update Plug-in Configurations utility
- Set PI AF server utility
- Capture AF SDK event trace output
- Track PI AF changes with Audit Trail
- View installed plug-ins
- Command-line plug-in registration
- Create an XML registration file
- Create an SQL registration script
- Register plug-ins with generated SQL scripts
- Plug-in provider management

Launch PSE with command line options

The PI System Explorer (PSE) can be invoked with command line options that control its initial selection. The PI System Explorer application is named **`AFExplorer.exe`** and is located in the `\PIPC\AF` folder.

Procedure

1. Open a Windows command window and change to the `\PIPC\AF` folder.
2. Type:
   ```
   afexplorer parameter=paramValue
   ```
   where *`parameter`* is one of the following three parameters:
   - **`/system`**
   - **`/database`**
   - **`/navigator`**

   To display a list of available parameters, type:
   ```
   afexplorer /?
   ```

AFExplorer parameters

The following table lists the available parameters for the **`AFExplorer`** utility.

Parameter	Description	Example
/system	Sets the system parameter to the hostname of the PI AF server to which PSE should connect by default.	`afexplorer /system=MyAFServer`
/database	Sets the database parameter to the name of the PI AF database that PSE should open initially.	`afexplorer /database=MyAFDatabase`
/navigator	Sets the navigator parameter to the browser plug-in that should be selected initially in PSE.	`afexplorer /navigator=Elements`

Navigator parameter values

The following table lists the available values for the **navigator** parameter.

Parameter value	Example
`Elements`	`afexplorer /Navigator=Elements`
`EventFrames`	`afexplorer /Navigator=EventFrames`
`Library`	`afexplorer /Navigator=Library`
`UnitOfMeasure`	`afexplorer /Navigator=UnitOfMeasure`
`Analyses`	`afexplorer /Navigator=Analyses`
`MyPI`	`afexplorer /Navigator=MyPI` **Note:** Use only if legacy notifications are present on a PI AF client computer.
`Notifications`	`afexplorer /Navigator=Notifications` **Note:** Use only if legacy notifications are present on a PI AF client computer.
`AFContactNavigator`	`afexplorer /Navigator=AFContactNavigator`
`Management`	`afexplorer /Navigator=Management`

PI AF Diagnostics utility

The PI AF Diagnostics (**AFDiag**) utility is a command-line utility that you can use to enable or disable PI AF server features and perform other administrative functions. The utility makes a direct connection with the associated SQL Server database and requires the SQL Server `sysadmin` or `db_AFadmin` role.

AFDiag is located in the `\PIPC\AF` folder.

Topics in this section

- Grant permissions for PI AF Diagnostics utility
- Run the AFDiag utility
- AFDiag utility parameters
- Audit Trail implementation

Grant permissions for PI AF Diagnostics utility

To use the PI AF Diagnostics utility, you need to grant the `db_AFadmin` database role to the SQL Server Login.

Procedure

1. In the Microsoft SQL Server Management Studio, connect to the SQL Server instance in which the PIFD database resides.
2. Under the SQL Server instance, expand the **Security** folder; then expand the **Logins** folder.
3. Right-click the login that corresponds to the appropriate Windows user and select **Properties**.
4. Select the **User Mapping** page.
5. Select the row for the PIFD database.
6. Select the **Map** check box for the PIFD database.
7. With the database still selected, select the **db_AFadmin** database role check box, as shown in the following figure.

8. Click **OK** to save your changes and close the Microsoft SQL Server Management Studio.

Run the AFDiag utility

You run the **`AFDiag`** utility in an elevated Windows command prompt window.

Procedure

1. As an administrator, open a Windows command prompt and change directory to `\PIPC\AF`.
2. Choose one of the following actions.

To ...	Do this ...
Display current configuration settings	Type `afdiag` with no parameter specified. A list of current settings is displayed for the PI AF server, database, Active Directory, and other configuration items such as audit trail, file extensions, and maximum size for files attached to PI AF objects.
Display a list of available parameters	Type `afdiag /?`

Execute a specific configuration task	Type `afdiag parameter` where *parameter* is one of the parameters listed in AFDiag utility parameters.

AFDiag utility parameters

Parameters for the **AFDiag** utility are listed in alphabetical order below.

- **/ActiveDirectory:*string* or /AD:*string***

  Tests access to Active Directory using the currently configured settings in the PI AF server. An optional user account can be specified to test another account. If an account is specified or set in the PI AF server settings, you also need to specify the **Password** parameter.

- **/AddIdentity:*string* or /AI:*string***

  Adds a security identity with the specified permissions to all security strings in the system. For example, to allow read and write access for *IdentityName* to all security strings, enter:
  `/AI:"IdentityName:A(r,w)"`

  If the specified security identity does not exist, it is created. Use PI System Explorer to create and/or configure mappings to the new identity.

- **/AuditTrailCleanupAdd:*string* or /ATCA:*string***

  Adds the specified user to the audit trail cleanup list. The audit cleanup job deletes audit records associated with this user from the PI AF SQL Server database.This parameter may be specified more than once.

- **/AuditTrailCleanupRemove:*string* or /ATCR:*string***

  Removes the specified user from the audit trail cleanup list. The audit cleanup job no longer deletes audit records associated with this user from the PI AF SQL Server database.This parameter may be specified more than once.

- **/CertificateAdd or /CA**

  Adds the specified client certificate to the PI AF SQL Server database. Use the **Password** parameter to specify a password for the certificate, if required.

- **/CertificateList or /CL**

  Lists the client certificates stored in the PI AF SQL Server database.

- **/CertificateRemove or /CR**

  Removes a client certificate from the PI AF SQL Server database by specifying the name of the certificate.

- **/CertificateSet:*string* or /CS:*string***

  Sets the server certificate in the PI AF server configuration file to the specified file. Use the **Password** parameter to specify a password for the certificate, if required.

- **/ChangeID:*string* or /CID:*string***

  Changes the ID for the PI AF server to the specified GUID.

- A return value of 1 means that the configuration change will be delayed.
- A return value of 2 means that you must restart the PI AF server.
- A negative return value indicates an error has occurred.

- **`/ClearChangeTables or /CCT`**

  Clears the **`findChanges`** and **`afdiag`** tables, which record information on changes to the system.

- **`/DeleteAuditTrail or /ATD`**

  Disables audit trail feature and deletes audit trail records in the PI AF SQL Server database. This operation permanently deletes the audit trail records, and requires `sysadmin` privileges. The audit trail records cannot be recovered if a delete is performed.

- **`/DeleteCases:`*`string`*` or /DelC:`*`string`***

  Deletes cases from the PI AF SQL Server database with a start time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

  - To specify a time, include a time string in **`hh:mm:ss`** format.
  - To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
  - To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

  For example, to delete cases that start between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelC:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

- **`/DeleteEventFrames:`*`string`*` or /DelEF:`*`string`***

  Deletes event frames from the PI AF SQL Server database with an end time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

  - To specify a time, include a time string in **`hh:mm:ss`** format.
  - To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
  - To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

  For example, to delete event frames that end between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelEF:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

  > **Note:**
  > Child event frames are also deleted, even if they do not match the criteria.

- **`/DeleteTransfers:`*`string`*` or /DelTR:`*`string`***

  Deletes transfers from the PI AF SQL Server database with an end time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

- To specify a time, include a time string in **`hh:mm:ss`** format.
- To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
- To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

For example, to delete transfers that end between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelTR:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

- **`/EnableAuditTrail or /AT`**

  Enables audit trail feature for the PI AF SQL Server database.

- **`/EnableAuditTrailCleanup[-] or /ATC[-]`**

  Enables audit trail cleanup feature for the PI AF SQL Server database. Requires `sysadmin` privileges.

- **`/EnableExternalDataTables[-] or /DT[-]`**

  Enables support for external PI AF tables.

- **`/EnablePropagationOfTargetDeletion[-] or /PTD[-]`**

  Enables support for propagating the deletion of targets (elements) to the referencing analyses and notifications.

- **`/ExeFile:`*`string`*` or /F:`*`string`***

  The path to the PI AF server executable file. Default value is `AFService.exe`.

- **`/ExternalDataTablesAllowNonImpersonatedUsers[-] or /DTImp[-]`**

  Enables support for external PI AF tables for non-impersonated users.

- ***`@file`***

  Reads response file for more options. The response file must contain one parameter per line. Comment lines start with the '#' character.

- **`/FileExtensions:`*`string`*` or /FE:`*`string`***

  Defines the types of file objects that can be attached to a PI AF file object, such as an annotation. The following file types are supported:

  - MS Office: csv, docx, pdf, xlsx
  - Text: rtf, txt
  - Image: gif, jpeg, jpg, png, svg, tiff
  - ProcessBook: pdi

  Enter the extensions as a colon-separated list, for example:

  ```
  /FE:docx:xlsx:csv:pdfd:pdi:jpg:png
  ```

- **`/FileExtensionsAdd:`*`string`*` or /FEA:`*`string`***

  Adds an additional file extension to the list of allowed PI AF file object types. See the list of supported file types above.

You can specify this parameter more than once.

- **`/FileExtensionsRemove:`*`string`*` or /FER:`*`string`***

  Removes a file extension from the list of allowed PI AF file object types. See the list of supported file types above.

  You can specify this parameter more than once.

- **`/FileMaxLength:`*`integer`*` or /FML:`*`integer`***

  Defines the maximum size in megabytes of a PI AF file object, such as an annotation. A value of zero disables support for all files. The default maximum allowed file size is 10MB.

- **`/FindDuplicates or /FD`**

  Searches for duplicate entries in all object tables and returns a list of tables. It identifies how many duplicates were found in each object table and logs them in an XML file. OSIsoft recommends that you make a backup of the PI AF SQL Server database (PIFD) before you proceed to delete the duplicate entries.

- **`/NewID or /NID`**

  Generates a new ID for the PI AF server.

  - A return value of 1 means that the configuration change will be delayed.
  - A return value of 2 means that you must restart the PI AF server.
  - A negative return value indicates an error has occurred.

- **`/Password or /PWD`**

  Specifies a certificate password for the `ActiveDirectory`, `CertificateAdd`, `CertificateSet`, or `PasswordEncryptionCertificate` options.

- **`/PasswordEncryptionCertificate:`*`string`*` or /PEC:`*`string`***

  Replaces the default password encryption for external database passwords with a custom certificate file. Use the **`Password`** option to specify a password for the certificate, if required.

- **`/PlugInVerifyLevel:`*`level`*` or /VL:`*`level`***

  Configures the level of verification required for plug-ins to run. Valid levels are:

  - **`None`**

    Disables validation; runs all plug-ins.

  - **`AllowUnsigned`**

    Runs unsigned plug-ins and plug-ins with valid signatures.

  - **`RequireSigned`**

    Runs only plug-ins with valid signatures.

  - **`RequireSignedTrustedProvider`**

    Runs only plug-ins with a valid signature from a trusted provider.

- **`/Port:integer or /P:integer`**

  Tests the specified port to the PI AF server. This is used to perform a basic port test to see if specified port on the computer used by the PI AF server can be opened and something is listening for a connection on the port. The standard ports used by the PI AF server are 5457 and 5459. This is similar to attempting to test a port using Telnet. It tests all IP addresses for the computer (both IPv4 and IPv6 addresses). Typically, you would test both ports 5457 and 5459.

- **`/RebuildPathCache or /RPC`**

  Rebuilds the path cache to each element in the PI AF SQL Server database. The path cache can become outdated after significant data insertions, edits, and/or deletions. Rebuilding the path cache can improve the PI AF server's performance.

  > **Note:**
  > The stored procedure used to rebuild the path cache is not supported on a secondary PI AF server (a subscriber) in a PI AF collective.

- **`/Reindex or /RI`**

  Completely rebuilds every index in the PI AF SQL Server database. This substantially improves the PI AF server's performance after a massive data insertion.

- **`/ResetAdministrator:string or /SRA:string`**

  Resets the permissions on the `Administrators` identity to allow full privileges on the local PI System. This parameter requires `sysadmin` or `db_owner` privileges.

  The default `/sra` enables privileges for the `BUILTIN\Administrators` account.

  `/sra:domainName\username` enables privileges for a particular user account.

- **`/Silent[-] or /S[-]`**

  Runs silent mode and prevents message display.

- **`/TrustedProviderAdd:providername or /PA:providername`**

  Adds the specified provider to the list of trusted plug-in providers. To add providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/TrustedProviderList or /PL`**

  Displays a list of trusted plug-in providers. To list providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/TrustedProviderRemove:providername or /PR:providername`**

  Removes the specified provider from the list of trusted plug-in providers. To delete providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/UomCaseSensitive[-] or /UCS[-]`**

  Changes configuration of UOM abbreviations from case insensitive to case sensitive in the PI AF SQL Server database. When enabled, only a 2015 R2 (v2.7.5) or later PI AF Client can connect.

  For more information, see Configuration of case-sensitive UOM abbreviations.

- **`/UpgradeAuditTrail or /ATU`**

  Upgrades audit trail records in the PI AF SQL Server database. Supports upgrades from PI AF 2.6 or later.

- **`/Version or /V`**

  Displays version information.

Audit Trail implementation

You can use the audit trail feature to examine audit trail records in the PI AF SQL database. Once enabled, audit trail records are created using SQL Server Agent jobs. You can view the audit trail with Process Explorer.

Enable audit trail

To enable the audit trail feature, you use the **`AFDiag`** utility and the **`EnableAuditTrail`** (**`/AT`**) parameter. For example:

```
afdiag /AT
```

Disable audit trail

To disable the audit trail feature and delete all audit trail records permanently, you must have `sysadmin` privileges on the SQL Server. To disable the audit trail feature, you use the **`AFDiag`** utility and the **`DeleteAuditTrail`** (**`/ATD`**) parameter. For example:

```
afdiag /ATD
```

Once you use this parameter, the audit trail is not recoverable.

Requirements

The audit trail feature uses SQL Server Change Data Capture (CDC) to generate an audit trail. CDC is supported in the following versions of SQL Server:

Version	Standard	Enterprise	Developer [1]	Evaluation [1]
SQL Server 2008	✗	✓	✓	✓
SQL Server 2008 R2	✗	✓	✓	✓
SQL Server 2012	✗	✓	✓	✓
SQL Server 2014	✗	✓	✓	✓
SQL Server 2016 [2]	✗	✓	✓	✓
SQL Server 2016 SP1 and later [2]	✓	✓	✓	✓

[1] Development system only.

[2] Only supported in PI AF 2016 and later. For more details on CDC support, see the OSIsoft Tech Support article KB01410 - PI AF error: Could not find stored procedure sys.sp_cdc_parse_captured_column_list (https://techsupport.osisoft.com/Troubleshooting/KB/KB01410).

The audit trail feature has the following additional requirements:

- The SQL Server Agent must be running before you enable the audit trail feature.
- You must be a member of the sysadmin role on the SQL Server that contains the PI AF SQL database.

Previous versions

The audit trail feature that was released in PI AF 2.6.0 or later is supported by this installation and is upgraded when the PI AF SQL scripts are executed. At that time, the audit trail records are upgraded to the current format. If for some reason a failure occurs during the upgrade, you can use the **AFDiag UpgradeAuditTrail** (**/ATU**) parameter to fix the audit trail tables and records.

Versions of the audit trail feature prior to PI AF 2.6.0 have a different format and are no longer updated with new change records after an upgrade of the PI AF SQL database. Existing tables and data remain intact, but new records cannot be added.

Audit trail support in PI AF collectives

The audit trail feature is not supported on secondary members of PI AF collectives. If the audit trail feature is currently enabled on a server that you wish to add as a secondary server to a PI AF collective, you must run the **AFDiag** utility and disable the feature. Audit trail data is only stored on the primary member of a PI AF collective and is not replicated to any secondary member of a PI AF collective.

Note:
When you are designing SQL Server backup procedures for your PI AF data, please be aware that if the primary member of a PI AF collective becomes unavailable, the audit trail data will also be unavailable. If the primary member of a PI AF collective cannot be recovered from a backup, the audit trail data cannot be recovered either.

Re-enabling audit trail

If you add an existing PI AF server with audit trail enabled to a collective as a secondary member, but later remove that server from the collective, you need to re-enable the audit trail feature on that particular server. Otherwise, auditing remains disabled.

AF Update Plug-in Configurations utility

The **AFUpdatePlugInConfigurations** utility provides a **Repair**, a **CreateConfig**, and a **ReplacePIServer** feature that enable you to perform bulk updates of attribute configuration strings with a single command. After you run the utility, click Refresh in PI System Explorer to see the changes.

The following table lists the available parameters for the **AFUpdatePlugInConfigurations** utility. Only one feature can be specified at a time in combination with the **/Root**, **/List**, and **/EventFrames** parameters.

Parameter	Description
/Root:*string*	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. Specifies the PI AF server or database on which to operate. Enclose the entire parameter string in quotation marks (" ").

Parameter	Description
/Repair[-]	Default parameter if no parameter is specified. For attributes on the PI AF server or database specified in the **/Root** parameter, corrects PI Point data reference attributes for which the stored configuration string has become out of synchronization with the PI Data Archive. This can occur for the following reasons: • Deleted PI points: When PI points are deleted and then recreated with the same name, the ID of the new point does not match the ID in the stored configuration string. • Renamed PI points: When PI points are renamed, the stored configuration string still uses the old PI point name. • Unresolved attributes: If the PI point to which a data reference points is not yet created, the stored configuration string does not contain the point ID.
/CreateConfig [-]	For attributes on the PI AF server or database specified in the **/Root** parameter, creates the PI point if it does not already exist, or updates it with any changes. This is the same operation as when you right-click an element or attribute in PI System Explorer and choose **Create or Update PI Point**.
/ReplacePIServer: *string*	For PI Point data reference attributes on the PI AF server or database specified in the **/Root** parameter, redirects attributes to a different PI Data Archive. Use a colon (**:**) to precede the existing PI Data Archive name. Separate the existing PI Data Archive name from the new PI Data Archive name with a semi colon (**;**) . Short form: **/Replace**
/List [-]	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. For the PI AF server or database specified in the **/Root** parameter, lists all attributes to be operated on.
/EventFrames:*string*	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. Performs specified operation on each attribute of all event frame templates and event frames that occurred between the start and end time specified, format, using standard international date notation YYYY-MMDD. Use a semi colon to separate the start and end time. Short form: **/EF**
/? or **/help**	Displays list of parameters
@*file*	Uses the specified file to provide additional input arguments. The file should contain one argument per line. Comment lines start with the '**#**' character.

/Repair syntax example

To repair stored configuration strings in the specified PI AF database so that they correctly map to the PI points on the PI Data Archive, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /Repair
```

/CreateConfig syntax example

To perform the CreateConfig operation in bulk on all attributes in a PI AF database, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /CreateConfig
```

To perform the CreateConfig operation on all attributes on a specific PI AF server, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer" /CreateConfig
```

/ReplacePIServer syntax example

To redirect all PI point data reference attributes in a specified PI AF database to a new PI Data Archive, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase"
   /ReplacePIServer:OldPIDataArchive;NewPIDataArchive
```

/EventFrames syntax example

To perform the CreateConfig operation in bulk on all event frame attributes in a PI AF database over a specific time range, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /EF:"2017-01-31
23:00:00";"2017-06-21 16:00:00"  /CreateConfig
```

Set PI AF server utility

The SetPISystem utility (**setpisystem**) enables you to configure known PI AF servers. **setpisystem** is located in the PIPC\AF directory.

SetPISystem utility parameters

The following table lists the available parameters for the SetPISystem utility.

Parameter	Short form	Description
/Name:*string*	**/N**	Specifies the name of the PI AF server to modify or create. If not specified, the default PI AF server is used.
/Host:*string*	**/H**	Specifies the hostname for the PI AF server.
/Protocol:*Tcp*|*NamedPipe*	**/C**	Specifies the protocol for the PI AF server.
/Port:*integer*	**/P**	Specifies the port for the PI AF server.
/Timeout:*integer*	**/T**	Specifies the timeout for the PI AF server in seconds.
/AccountName:*string*	**/A**	Specifies the account name for the PI AF server.
/DefaultPISystem	**/D**	Sets the specified PI AF server as the default PI AF server.
/Remove	**/R**	Removes the specified PI AF server from the list of known PI AF servers.
/Silent[-]	**/S**	Establishes silent mode, which prevents message display.
/List	**/L**	Lists the current known PI AF servers.
/AddAlias:*string*	**/AA**	Adds the specified alias to the PI AF server.
/RemoveAlias:*string*	**/RA**	Removes the specified alias from the PI AF server.

Capture AF SDK event trace output

You use the **AFGetTrace** utility (**afgettrace.exe**) to capture event trace output from the AF SDK. Event tracing can help you debug an application and perform capacity and performance analysis.

Procedure

1. Open a command window and change directory to PIPC\AF.
2. Choose from the following actions:

To ...	Do this ...
Display syntax and parameters	At the command prompt, type: `afgettrace /?`
Run **AFGetTrace** with default settings	At the command prompt, type: `afgettrace` Default output goes to standard output.
Run **AFGetTrace** with specific parameters	At the command prompt, type: `afgettrace /parameter` Refer to AFGetTrace utility parameters for details on the parameters you can use.
Terminate event tracing	In the command window, type: `X` **Note:** If you close the command window without terminating **afgettrace**, trace events continue to be generated, which can slow down your AF SDK applications.

AFGetTrace utility parameters

The following table lists available parameters for the **AFGetTrace** utility.

Parameter	Short form	Description
/Provider:*string*	**/P**	Specifies the name of the event tracing session. You only need to specify to use an existing event trace provider. The default value is AFGetTrace.
/Level:{Critical|Error|Warning|Warning|Information|Verbose|Detail}	**/L**	Specifies the level of detail to be included in the events written by the AF SDK. Detail at or above severity of the level chosen is generated. The default value is Verbose.
/Keywords:{None|Server|Connection|Cache|Events|Trace|Data|All}	**/K**	Specifies the keywords used to determine the category of events that you want the AF SDK to write. The AF SDK writes the event if any of the event's keywords match the keywords specified in this setting. The default value is All.

Parameter	Short form	Description
/EnableAF[-]	**/AF**	Enables messages from the AF SDK message provider. The default value is `True`.
/EnablePI[-]	**/PI**	Enables messages relating to communication with the PI Data Archive from the MDA message provider. The default value is `True`.
/LogFile:*string*	**/Log**	Specifies the name of the log file for trace output messages. Messages are still displayed on standard output unless you specify the **Silent** parameter.
/LogFileMaxSize:*double*	**/FileMax**	Specifies the maximum size of the log file in megabytes. The trace ends once the size is reached. Enter `0` for no limit. The default value is `0`.
/Timeout:*integer*	**/TO**	Specifies the number of minutes before the trace ends. Enter `0` for no timeout. The default value is `0`.
/Mask:*string*	**/M**	Prevents any message containing the specified mask from being displayed. Enclose the mask in quotes if it contains spaces. This parameter can be specified more than once, but values must be unique.
/MaskPID:*UInt32*	**/MPID**	Prevents any message associated with the specified Process ID from being displayed. This parameter can be specified more than once. **Note:** This parameter is ignored if the **ProcessID** parameter is specified.
/MaskTID:*UInt32*	**/MTID**	Prevents any message associated with the specified Thread ID from being displayed. This parameter can be specified more than once. **Note:** This parameter is ignored if the **ThreadID** parameter is specified.
/NoHeader[-]	**/NH**	Disables header information on each message. Header information includes the time stamp, process identifier, and thread identifier.
/ProcessID:*UInt32*	**/PID**	Displays any message associated with the specified process ID. This parameter can be specified more than once.
/ThreadID:*UInt32*	**/TID**	Displays any message associated with the specified thread ID. This parameter can be specified more than once.
/WordWrap	**/W**	Word wraps output messages to the width of the console.
/Silent[-]	**/S**	Establishes silent mode, which prevents message display.

Track PI AF changes with Audit Trail

If Audit Trail is enabled for your system, users with administration privileges can use the utility to view changes to PI AF objects from all PI AF databases, or to a single object.

Beginning with PI AF 2017 R2, users with administration privileges can right-click an object in the browser or an object on a list in the viewer and click **Audit Trail Events** to review audit data specific to that object only.

See Audit Trail implementation for details about enabling the Audit Trail feature.

Each row in the table of the AF Audit Trail window contains data that identifies a specific change to a PI AF object. You can double-click a row to view details about that change in the AF Audit Trail Details window.

You can press CTRL+C and CTRL+V to copy and paste rows from either window into another document, such as a spreadsheet.

Review changes with the Audit Trail utility

Use the Audit Trail utility to track changes to one or more PI AF objects.

Before you start

- Only users with administrator privileges can use the Audit Trail utility.
- Both the PI AF client and PI AF server should be located in the same time zone for audit trail queries to work properly.

Procedure

1. Choose from the following actions:

To ...	Do this ...
Review changes to PI AF objects in all databases	Click **Tools** > **Audit Trail**.
Review changes to a single PI AFobject	a. Open the database that contains the object you want to audit. b. In the navigator, click the object type you want to audit. For example, to audit a table, click **Library**. c. In the browser, expand the tree until the object you want to audit is displayed in either the browser tree or listed in the viewer. d. Right-click the object and click **Audit Trail Events**.
Review changes to the UOM database	a. In the navigator, click **Units of Measure**. b. Click **File** > **Audit Trail Events**.

2. In the table displayed in the AF Audit Trail window, review changes made to PI AF data. The following columns are displayed:

Column	Description
Date	The date and time of the change.
Action	The type of change (Insert, Update, or Delete).
Type	The type of object that changed.
Database	The PI AF database containing the changed object.
Path	The hierarchical path to the changed object (when the object is a child of a parent object, the path shows the parent object).

Column	Description
Name	The name of the changed object.
User	The user who made the change, in the form of *domain\user*.

a. You can modify the content of the table, as follows:

To ...	Do this ...
Adjust the time period	a. Change **Start Time** and **End Time** to adjust the time period. You can choose from the following actions: ▪ Click to choose or enter a new date and/or time. ▪ Click to construct a relative time expression. b. Click (or press Enter) to display the updated table.
View changes for a subsequent time period	Click .
View changes for a previous time period	Click .
Adjust the number of change rows for current time period	a. Enter the number of rows you want to view in the **Maximum** field. b. Click (or press Enter) to display the updated table. **Note:** You can use **Maximum** to control response time. In addition, if the number of changes in the current time period exceeds **Maximum**, you can increase its value to display them.
Filter the returned results	a. Click to select a filter type. b. Enter text in the **Filter** field. c. If necessary, click . The table displays only those rows that contain your filter criteria.

3. Optional. To view details for a specific row, double-click it.

   a. In the AF Audit Trail Details window, review the details of the row you selected. In addition to **Path\Name**, **Indentity** and **Date**, the following information is displayed:

Column	Description
Action	Type of change (Insert, Update, or Delete) and the PI AF object sub-object (such as an attribute or attribute value).
Name	The PI AF sub-object; can be blank.
Id	The identity of the object referenced in the detail record. The value is either a GUID or an integer. The identity can be useful in situations where an object is renamed, since the underlying identity does not change.

Column	Description
Property Name	The column name from the SQL table in PI AF SQL Database (PIFD) for the changed sub-object. While this name is not necessarily easily understood, it provides a method for including all data within the record that has been changed.
Old Value	The value before the change.
New Value	The value after the change.

b. Click **Close** to return to the AF Audit Trail window.

4. Optional. You can export selected data grid rows or all the rows from the time range (if no rows are selected) in CSV format:
   a. In the **Export to File** field, enter a name for the export file.
   b. Click ... to open the Save As window and navigate to a specific directory.
   c. If you want to export the data from Audit Trail Details window for each selected row, select the **Detailed** check box.
   d. Click **Export All** or **Export Selected**, depending on whether you selected specific rows. (A tooltip indicates the number of rows selected.) The status bar initially indicates the total number of rows and then appends the time when the export completed. Note that while the export is executing, you can cancel it by clicking **Cancel Export**. The status bar indicates the time when the export was canceled.
5. To exit the Audit Trail utility, click **Close**.

View installed plug-ins

You check on installed plug-ins in the PI AF Server Properties window.

Procedure

1. Select **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Plug-Ins** tab.
   - Move the pointer over a plug-in to see whether it is loaded and the version loaded.
   - Right-click a plug-in and select **Properties** to view details, such as the assembly, version, loaded version, support assemblies, and so on.

> **Note:**
> The loaded plug-in version usually matches the version on the PI AF server. However, if the version on the server changes, you must restart PI System Explorer to load the new plug-in version on the server.

Command-line plug-in registration

You can register different Plug-In implementations and support assemblies on the command line, using the **`RegPlugIn`** and **`RegPlugIn64`** utilities. For a list of parameters, see RegPlugIn parameters.

The Plug-In registration utilities are located in the `\PIPC\AF` folders for their corresponding `Program Files` directories:

```
\Program Files\PIPC\AF\RegPlugIn64.exe
\Program Files (x86)\PIPC\AF\RegPlugIn.exe
```

Examples

In the following examples, *MyPISystem* is the PI AF server where the Plug-In is to be registered, *MyPlugIn.dll* is the name of the Plug-In assembly, and *Support.dll* is the name of a support assembly for the Plug-In.

- .NET 3.5 Only Plug-In Targeting Any CPU

  The simplest to register, this implementation works on any operating system architecture and both versions of the .NET Framework. It cannot use new features that are specific to the .NET 4 version of AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll
  ```

- .NET 4 Only Plug-In Targeting Any CPU

  This implementation works on any operating system architecture and the .NET 4 Framework. It does not work with the .NET 3.5 version of the AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\4.0\MyPlugIn.dll
  ```

- .NET 3.5 and .NET 4 Plug-In Targeting Any CPU

  This Plug-In has two implementations. One targets the .NET 3.5 Framework and works with the .NET 3.5 version of AF SDK, and the other targets the .NET 4 Framework and works with the .NET 4 version of AF SDK. Both implementations are registered with the following command:

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\4.0\MyPlugIn.dll
  ```

- .NET 3.5 Only Plug-In Targeting 32- and 64-Bit Platforms

  This Plug-In has two implementations for the .NET 3.5 Framework. One implementation targets 32-bit operating systems and the other one targets 64-bit operating systems. With both implementations registered, the Plug-In works on any operating system architecture and both versions of the .NET Framework, but cannot use the new features that are specific to the .NET 4 version of AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\x86\MyPlugIn.dll
  ```

  ```
  RegPlugIn64 /PISystem:MyPISystem PlugIns\x64\MyPlugIn.dll
  ```

- .NET 3.5 and .NET 4 Plug-In Targeting 32- and 64-Bit Platforms

  This Plug-In has four implementations. Two implementations are for the .NET 3.5 Framework, one for 32- and one for 64-bit platforms. The other two implementations are for the .NET 4, one for 32- and one for 64-bit platforms. With all implementations registered, it works on any operating system architecture and both versions of the .NET Framework.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\x86\MyPlugIn.dll PlugIns
  \4.0\x86\MyPlugIn.dll
  ```

  ```
  RegPlugIn64 /PISystem:MyPISystem PlugIns\x64\MyPlugIn.dll PlugIns
  \4.0\x64\MyPlugIn.dll
  ```

- .NET 3.5 Only Plug-In and Support Assembly Targeting Any CPU

  This implementation, with its support assembly, works on any operating system architecture and both versions of the .NET Framework. It cannot use the new features that are specific to the .NET 4 version of AF SDK.

```
RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\Support.dll
```

- .NET 3.5 and .NET 4 Plug-In and Support Assembly Targeting Any CPU

  This Plug-In has two implementations with two implementations of its support assembly. One targets the .NET 3.5 Framework and works with the .NET 3.5 version of AF SDK. The other targets the .NET 4 Framework and works with the .NET 4 version of AF SDK. Both implementations along with their support assemblies are registered with the following command.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\Support.dll PlugIns
  \4.0\MyPlugIn.dll PlugIns\4.0\Support.dll
  ```

RegPlugIn parameters

The following table lists available parameters for the **RegPlugin** utility.

Parameter	Short form	Description
/PISystem:*string*	**/P**	Specifies the PI AF server to process. If not specified, the default PI AF server is used.
/OutputFile:*string*	**/O**	Specifies an output file. If specified, a SQL script is generated to create Plug-Ins. Use the **AppendFile** parameter to append to an existing file.
/AppendFile[-]	**/A**	Used in conjunction with the **OutputFile** parameter, specifies whether to append to an existing output file.
/List	**/L**	Lists registered assemblies in the specified PI AF server.
/Recursive[-]	**/R**	Automatically processes support assemblies in subdirectories based upon assembly and directory names. Use the **RootDirectory** parameter to specify a root directory that is different than the current working directory.
/Force[-]	**/F**	Forces registration of assemblies that are older than currently registered assemblies or forces the deregistration of assemblies that do not appear to be registered.
/Silent[-]	**/S**	Silent mode. Prevents messages from being displayed.
/Unregister	**/U**	Removes assembly and plug-ins from the specified PI AF server.
/RegFile:*string*	**/RF**	Creates a registration XML file for the specified assembly that can be used to register a plug-in assembly and all its related support assemblies. Any additional files specified are treated as support assemblies.
/Version	**/V**	Displays version information. All other parameters are ignored.
/User:*string*	**/user**	Specifies a different Windows user account to connect to the PI AF server.
/Password:*string*	**/PW**	Used in conjunction with the **User** parameter, specifies the password of a different Windows user account to connect to the PI AF server.

Parameter	Short form	Description
/Owner:*string*	**/Own**	Provides the owner file name of all specified support assemblies. Normally used with the **Directory** parameter when registering support assemblies. By default, the directory name will be determined relative to the **RootDirectory** parameter or current working directory if the root directory is not specified.
/Directory:*string*	**/Dir**	Provides the directory name for all specified support assemblies. Normally used with the **Owner** parameter when registering support assemblies. By default, the directory name is determined relative to the **RootDirectory** parameter or current working directory if the root directory is not specified.
/RootDirectory:*string*	**/RootDir**	Registers all assemblies relative to the specified root directory name. Normally used with the **Recursive** parameter when registering assemblies instead of using the **Directory**parameter. By default this parameter is set to the current working directory.
/Exclude:*string*	**/E**	Specifies input files or directories to exclude when searching a directory for files. You can specify this parameter more than once, but values must be unique.
files		Specifies the input assembly files, registration files, or directories to process. You can use the following wildcard specifiers to filter files processed in directories: • '*' matches zero or more characters • '?' matches exactly one character You can also use the **Exclude** parameter to filter the processed files.
@*file*		Provides additional input parameters to the specified file. The file should contain one parameter per line. Comment lines start with the '#' character.
/?, **/help**		Displays these option descriptions.

Create an XML registration file

For complex or frequently-executed registrations, you can create an XML file that contains the required settings.

Procedure

- To create the XML file, invoke `RegPlugin`, specifying both the required settings and the XML file name using the **`/RegFile:`** (**`/RF:`**) parameter.

Example

To create an XML file that registers a .NET 3.5 and .NET 4.0 version of `MyPlugIn.dll`, issue the following command:

```
RegPlugIn /RF: PlugIns\MyPlugIn.dll PlugIns\4.0\MyPlugIn.dll
```

The resulting registration file can be edited to supply any additional information required for the registration of the plug-in (not normally necessary). For 64-bit plug-ins, the registration file must be edited to set `LoadProperties` to `x64` and ensure that `Directory` is set to `x64`:

```
<SupportAssembly>
  <FilePath>x64\AFDRTest32Bit64Bit.dll</FilePath>
  <ID>1e00000c-3228-366a-3809-737433324269</ID>
  <Name>AFDRTest32Bit64Bit</Name>
  <Description>AFDRTest32Bit64Bit.dll Support Assembly</Description>
  <Directory>x64</Directory>
  <LoadProperties>x64</LoadProperties>
</SupportAssembly>
```

To register both implementations using the registration file, issue the following command:

```
RegPlugIn /PISystem:MyPISystem MyPlugIn.xml
```

Create an SQL registration script

Procedure

- To create an SQL registration script, invoke the `RegPlugIn` utility, specifying the required details and the `/OutputFile` or `/AppendFile` parameter. The resulting script can be loaded into the SQL database using SQL Server Management Studio or executed from a command line using the **`sqlcmd`** utility, enabling you to install plug-ins on machines where the PI AF SDK is not installed.

  Note:

  For plug-in developers: The SQL script needs to be generated every time you compile a new version of the plug-in.

Example

For example, to create a script that registers a NET 3.5-only Plug-In targeting any CPU, issue the following command:

```
RegPlugIn /O:MyPlugIn.sql PlugIns\MyPlugIn.dll
```

To generate an SQL registration script named `MyPlugIn.sql` that registers two implementations of MyPlugIn, a .NET 3.5-only Plug-In targeting x86 and x64, issue the following commands:

```
RegPlugIn /O:MyPlugIn.sql PlugIns\x86\MyPlugIn.dll
RegPlugIn64 /A:MyPlugIn.sql PlugIns\x64\MyPlugIn.dll
```

To generate an SQL registration script from a previously-created XML registration file, specify the XML file name on the command line. For example:

```
RegPlugIn /O:MyPlugIn.sql MyPlugIn.xml
```

Register plug-ins with generated SQL scripts

You register a plug-in from Microsoft SQL Server Management Studio or from the command line.

Procedure

1. Launch Microsoft SQL Server Management Studio.
2. Choose **File** > **Open** > **File**
3. Browse to the script and load it.
4. Execute the script.
5. To run the script from the command line, invoke the **`sqlcmd`** utility, specifying the **`-i`** *inputfile* option with the path and name of the SQL script as *inputfile* and the connection settings required to connect to the database server.

Plug-in provider management

By default, you can run PI AF plug-ins from any provider. For increased security, you can configure PI AF so that only plug-ins from trusted providers are allowed to be loaded and executed in the client application (PI AF server 2.5 and higher). Plug-in providers are encouraged to code-sign their plug-ins using Authenticode technology. You control how plug-in security is enforced by setting the verify level.

Verification level

You can display the verification level used when plug-ins are loaded by issuing the **`afdiag`** command with no parameters. The verification level is listed in the Configuration Settings section of the resulting output as `PluginVerifyLevel`.

`/PlugInVerifyLevel` parameter

You set the verify level with the **`afdiag`** command, and specify the **`/PlugInVerifyLevel`** parameter. Valid levels are:

- `None`: Disable validation; run all plug-ins.
- `AllowUnsigned`: Run unsigned plug-ins and plug-ins with valid signatures.
- `RequireSigned`: Run only plug-ins with valid signatures.
- `RequireSignedTrustedProvider`: Run only plug-ins with a valid signature from a trusted provider.

Other **`AFDiag`** parameters for managing plug-in providers

The following **`afdiag`** parameters are available to manage plug-in providers:

- Display a list of trusted providers: **`/TrustedProviderList`**
- Add a trusted provider: **`/TrustedProviderAdd:`***providername*
- Remove a trusted provider: **`/TrustedProviderRemove:`***providername*

For more information on these parameters, see AFDiag utility parameters. You can locate the provider name by viewing the digital signature for the plug-in DLL, as described in Locate the names of trusted providers.

Locate the names of trusted providers

Determine the name of a trusted provider of a plug-in DLL by viewing the details of its digital signature. The string for the trusted provider name must be contained in the certificate's subject.

Procedure

1. Right-click the plug-in DLL and click **Properties**.
2. In the Properties window, click the **Digital Signatures** tab. (If the plug-in is unsigned, this tab is absent.)
3. Select a **Name of signer** in the **Signature list** and click **Details**.
4. In the Digital Signature Details window, click **View Certificate**.
5. In the Certificate window, click the **Details** tab and scroll to the **Subject** field.

   The value displayed for this field is the name of the trusted provider. You can you use any subset or the whole name of any of the names that are part of the subject as the *<providername>* variable for the **`/TrustedProviderAdd`** and **`/TrustedProviderRemove`** parameters.

Technical support and other resources

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or through the OSIsoft Tech Support Contact Us page (https://techsupport.osisoft.com/Contact-Us/). The website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Details about your computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

To ask questions of others who use OSIsoft software, join the OSIsoft user community, PI Square (https://pisquare.osisoft.com). Members of the community can request advice and share ideas about the PI System. The PI Developers Club space within PI Square offers resources to help you with the programming and integration of OSIsoft products.

PI Interface for IEEE C37.118

Version 1.1.0.x

OSIsoft, LLC
777 Davis St., Suite 250
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Fax: (01) 510-357-8136
Web: http://www.osisoft.com

OSIsoft Australia • Perth, Australia
OSIsoft Europe GmbH • Frankfurt, Germany
OSIsoft Asia Pte Ltd. • Singapore
OSIsoft Canada ULC • Montreal & Calgary, Canada
OSIsoft, LLC Representative Office • Shanghai, People's Republic of China
OSIsoft Japan KK • Tokyo, Japan
OSIsoft Mexico S. De R.L. De C.V. • Mexico City, Mexico
OSIsoft do Brasil Sistemas Ltda. • Sao Paulo, Brazil
OSIsoft France EURL • Paris, France

PI Interface for IEEE C37.118

Copyright: © 2007-2015 OSIsoft, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of OSIsoft, LLC.

OSIsoft, the OSIsoft logo and logotype, PI Analytics, PI ProcessBook, PI DataLink, ProcessPoint, PI Asset Framework (PI AF), IT Monitor, MCN Health Monitor, PI System, PI ActiveView, PI ACE, PI AlarmView, PI BatchView, PI Coresight, PI Data Services, PI Event Frames, PI Manual Logger, PI ProfileView, PI WebParts, ProTRAQ, RLINK, RtAnalytics, RtBaseline, RtPortal, RtPM, RtReports and RtWebParts are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Published: 08/2015

Table of Contents

Chapter 1. Introduction 1

Reference Manuals 4
Supported Operating Systems 4
Supported Features 5
Diagram of Hardware Connection 8

Chapter 2. Principles of Operation 9

Startup Process 9
LocalEndPoint 10
RemoteEndPoint 10
PMU 10
Loading PI Points 10
Data Processing 11
Data Flow to the PI System 12
Multi-Threading 12
Communications Plug-Ins 13
PIC37118_Comm001 13
PIC37118_Comm002 13
PIC37118_Comm003 13
Data Quality Indications 13
Interface Robustness 14
PDC / PMU Data Source Level Failover 14
Failover 15
Communication Failure 15
Detection of Stale Data 15

Chapter 3. Installation Checklist 17

Data Collection Steps 17
Interface Diagnostics 18
Advanced Interface Features 19

Chapter 4. Interface Installation 21

Naming Conventions and Requirements 21
Interface Directories 22
PIHOME Directory Tree 22
Interface Installation Directory 22
Interface Installation Procedure 22
Silent Installation Procedure 22
PI Trust for Interface Authentication 22
Installing Interface as a Windows Service 23
Installing Interface Service with PI Interface Configuration Utility 23
Service Configuration 24
Installing Interface Service Manually 29

Chapter 5. Digital States **31**

Chapter 6. PointSource **33**

Chapter 7. PI Point Configuration **35**

Point Attributes 35
Tag 35
PointSource 36
PointType 36
Location1 36
Location2 36
Location3 37
Location4 37
Location5 37
InstrumentTag 37
ExDesc 40
Scan 40
Shutdown 40
DataSecurity 41
PtSecurity 41

Chapter 8. Startup Command File **43**

Configuring the Interface with PI ICU 43
C37118 Interface page 46
Sample PIC37118 XML File 63
Command-line Parameters 64
Sample PIC37118.bat File 71

Chapter 9. UniInt Failover Configuration **73**

Introduction 73
Quick Overview 74
Synchronization through a Shared File (Phase 2) 75
Configuring Synchronization through a Shared File (Phase 2) 76
Configuring UniInt Failover through a Shared File (Phase 2) 79
Start-Up Parameters 79
Failover Control Points 81
PI Points 82
Detailed Explanation of Synchronization through a Shared File (Phase 2) 86
Steady State Operation 87
Failover Configuration Using PI ICU 89
Create the Interface Instance with PI ICU 89
Configuring the UniInt Failover Startup Parameters with PI ICU 90
Creating the Failover State Digital State Set 91
Using the PI ICU Utility to create Digital State Set 91
Using the PI SMT 3 Utility to create Digital State Set 92
Creating the UniInt Failover Control and Failover State Points (Phase 2) 95

Chapter 10. Interface Node Clock **97**

Chapter 11. Security **99**

Windows 99

Authentication 99
Authorization 100

Chapter 12. Starting / Stopping the Interface 103

Starting Interface as a Service 103
Stopping Interface Running as a Service 103

Chapter 13. Buffering for PI Interfaces 105

Buffering services 105
Buffering and collectives 105
Buffering configuration 105

Chapter 14. Interface Diagnostics Configuration 107

Scan Class Performance Points 107
Performance Counters Points 110
Performance Counters 112
Performance Counters for both (_Total) and (Scan Class x) 112
Performance Counters for (_Total) only 113
Performance Counters for (Scan Class x) only 116
Interface Health Monitoring Points 117
I/O Rate Point 122

Appendix A. Error and Informational Messages 125

Message Logs 125
Messages 126
Interface Configuration Error Messages 126
Point Loading Error Messages 128
Communications Related Error Messages 129
System Errors and PI Errors 131
UniInt Failover Specific Error Messages 132
Informational 132
Errors (Phase 1 & 2) 133
Errors (Phase 2) 134

Appendix B. PI SDK Options 135

Appendix C. Communication Plug-In Selection 137

Appendix D. Terminology 139

Interface Specific Terms 139
General Terms 139

Appendix E. Technical Support and Resources 143

Appendix F. Revision History 145

Chapter 1. Introduction

This document describes the use of the PI Interface for IEEE C37.118, from here on referred to as the PI C37.118 interface. The PI C37.118 interface supports the IEEE C37.118/D7.3.5 Standard for Synchrophasors for Power Systems.

The PI C37.118 interface operates with any C37.118 compliant device via an Ethernet or serial interface. When connected to the C37.118 or Phasor Measurement Unit (PMU) or Phasor Data Concentrator (PDC) the interface provides highly accurate measurement of electrical power quality. Each instance of the interface can connect to multiple PDCs or PMUs and provides electrical measurements to the PI System.

The IEEE C37.118 specification does not specify what electrical measurements the device must provide; rather, it specifies only how the protocol-specific data (if provided) must be formatted. Therefore, each C37.118 device may provide its own unique set of Phasor measurements. The PI C37.118 interface is capable of handling specific Phasor configuration sets that are provided by the C37.118 compliant device.

The following electrical measurements and flags are typically provided by C37.118 compliant devices:

- C37.118 compliant Phasor data calculated at a rate defined by the PDC or PMU.
    - System Frequency
    - Frequency Deviation
    - Frequency Rate of Change
    - A, C, B magnitude and phase angle for Voltage and Current given in 16-bit (scaled) Polar format
    - A, C, B real and imaginary parts of complex number for Voltage and Current given in 16-bit (scaled) Rectangular format
    - A, C, B magnitude and phase angle for Voltage and Current given in 32-bit IEEE floating point Polar format
    - A, C, B real and imaginary parts of complex number for Voltage and Current given in 32-bit IEEE floating point format
    - Analog channel data as a 16-bit integer (scaled) or 32-bit IEEE floating point. The values and ranges defined by user.
    - Digital channel data as a 16-bit word with values and ranges defined by the user.

- Quality information reflecting the accuracy of the C37.118 data is available via a series of optional quality tags.
    - Data Valid quality returned in the Data Valid flag (Bit 15) of the C37.118 STAT word indicates problems with the C37.118 data block. Some data in the block may be invalid or filled in.
    - PMU Error quality returned in the PMU Error flag (Bit 14) of the C37.118 STAT word indicates internal errors with the PMU such as A/D calibration errors, memory errors, etc.
    - PMU Sync Error quality returned in the PMU Sync flag (Bit 13) of the C37.118 word indicates a loss of external time synchronization such as the loss of satellite tracking or IRIG-B input failure.
    - Data Sorting information used by PDC's to indicate the method by which the PMU data is inserted into the data frame. This information is extracted from the Data Sorting flag (Bit 12) of the C37.118 STAT word.
    - Time Quality indicator codes extracted from the Time Quality field (Bits 3-0) of the C37.118 FRACSEC word can be used with the PMU Sync Error quality and indicates the accuracy of the timestamp provided.
    - Time Lock quality can be used in conjunction with the PMU Sync Error quality to determine the length of time that time sync has been lost. This information is extracted from the Unlocked Time flags (Bits 4 and 5) of the C37.118 STAT word.
    - Leap Second status can be extracted from the Leap Second flags (Bits 6-4) of the C37.118 FRACSEC word indicates if a leap second is pending or occurring and indicates the direction of the leap.
- Non C37.118 measurement information can also be stored in the PI Data Archive.
    - Interface diagnostic information
    - Longitude and Latitude of the PMU.

Measured data from the PMU is time-stamped by the PMU in accordance with the C37.118 standard for C37.118 data. Timestamp accuracy is achieved through an internal Global Positioning System (GPS) satellite receiver that synchronizes the PMU to within 1 µs of Coordinated Universal Time (UTC).

If a PDC or PMU indicates a data quality or time synchronization issue by setting any of the Data Valid, PMU Error, PMU Sync Error or Data Sorting Type flags in the STAT word, the interface can do one of the following (this behavior is controlled via the XML configuration file):

- Store normally.
- Mark all PMU measurements as Questionable.
- Apply a System Digital State corresponding to the flag(s) that are set. If multiple flags are set, the most critical applies.
- Discard the measurement values.

Multiple instances of the interface can be run on a single PI interface node. However, when running multiple instances of the interface on a single node, CPU and memory allocation should be monitored to ensure the system remains in a stable state of operation. The exact number of instances that may be run on a single PI interface node will vary according to hardware and network specifications.

Both PDC / PMU Data Source Level Failover and UniInt Phase 2 interface level failover are supported. When running in PDC / PMU Data Source Level Failover mode, the interface obtains data from one of two available sources PDC / PMU. The source PDC / PMU must have identical phasors, analog and digital channels and data streams for each interface. This requirement ensures that the interface obtains the same data regardless of which source is active. When running in UniInt Failover mode, two copies of the interface are connected to the PMU/PDC at the same time. When the primary interface stops collecting data, the backup interface assumes the primary role and continues data collection. PMU/PDC Data Source Level Failover and UniInt Phase 2 interface level failover modes can be run simultaneously. Failover maximizes interface data availability on the receiving PI Data Archive(s).

Note: It should be noted that the C37.118 specification is very vague with respect to communications configuration, and the various PDC / PMU vendors have implemented devices in a variety of ways. As such, the design of the C37118 interface uses an XML configuration file to provide the information needed to establish communications with the various C37.118 devices. **Check with OSIsoft to verify that your make and model of PDC or PMU has been validated for use with the C37118 interface.**

Note: The value of the [PIHOME] variable for the 32-bit interface will depend on whether the interface is being installed on a 32-bit operating system (`C:\Program Files\PIPC`) or a 64-bit operating system (`C:\Program Files (x86)\PIPC`).

The value of the [PIHOME64] variable for a 64-bit interface will be: `C:\Program Files\PIPC` on a 64-bit operating system.

In this documentation [PIHOME] is used to represent the value for either [PIHOME] or [PIHOME64]. The value of [PIHOME] is the directory in which PI client applications are stored.

Note: This interface has been built against a UniInt version (4.5.0.59 and later) which now writes all its messages to the local PI Message log.

Please see the document *UniInt Interface Message Logging.docx* in the `%PIHOME%\Interfaces\UniInt` directory for more details about how to access these messages.

Note: OSIsoft is revising product documentation and other literature to reflect the evolution of the PI Server from a single server to a multi-server architecture. Specifically, the original historian core of the PI Server is now referred to as the PI Data Archive.

Originally, the PI Server was a single server that contained the PI Data Archive and other subsystems. To add features and improve scalability, the PI Server has evolved from a single server to multiple servers. While the PI Data Archive remains a core server of the PI Server product, the product name "PI Server" now refers to much more than the PI Data Archive. OSIsoft documentation, including this user manual, is changing to use "PI Server" in this broader sense and "PI Data Archive" to refer to the historian core.

Reference Manuals

OSIsoft

- PI Data Archive manuals
- *PI API Installation Instructions* manual (%PIHOME%\bin\API_install.doc)
- *PI Universal Interface (UniInt) User Guide* (%PIHOME%\Interfaces\UniInt\PI Universal Interface (UniInt) User Guide.pdf)
- *UniInt Interface Message Logging for UniInt 4.5.0.x and later Interfaces* (%PIHOME%\Interfaces\UniInt\UniInt Interface Message Logging.pdf)
- *PI Interface Configuration Utility* User Guide (%PIHOME%\ICU\PI Interface Configuration Utility.pdf)

Vendor

- *IEEE C37.118/D7.3.5 Standard for Synchrophasors for Power Systems*

Supported Operating Systems

Platforms		32-bit application	64-bit application
Windows Vista	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 2008	32-bit OS	Yes	No
Windows 2008 R2	64-bit OS	Yes (Emulation Mode)	No
Windows 7	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 8	32-bit OS	Yes	No
	64-bit OS	Yes (Emulation Mode)	No
Windows 2012 Server	64-bit OS	Yes (Emulation Mode)	No

The interface is designed to run on the above mentioned Microsoft Windows operating systems and their associated service packs.

Please contact OSIsoft Technical Support for more information.

Security Note: OSIsoft recommends installing all available updates from the Windows Update service. OSIsoft also recommends using the newest versions of Windows to obtain the latest security features.

Supported Features

Feature	Support
Interface Part Number	PI-IN-OS-C37.118-NTI
Auto Creates PI Points	No
Point Builder Utility	No
ICU Control	Yes
PI Point Types	Digital / Int16 / Int32 / Float16 / Float32
Sub-second Timestamps	Yes
Sub-second Scan Classes	No (Unsolicited Only)
Automatically Incorporates PI Point Attribute Changes	Yes
Exception Reporting	No
Outputs from PI	No
Inputs to PI:	Unsolicited
Supports Questionable Bit	Yes
Supports Multi-character PointSource	Yes
Maximum Point Count	No
* Uses PI SDK	No
PINet String Support	No
* Source of Timestamps	C37.118 Datastream
History Recovery	No
* UniInt-based * Disconnected Startup * SetDeviceStatus	Yes Yes Yes
* Failover	Data Source Level Failover UniInt Failover (Phase 2 Cold and Hot)
* Vendor Software Required on Interface Node / PINet Node	No
Vendor Software Required on Foreign Device	No
Vendor Hardware Required	Yes
Additional PI Software Included with interface	No
Device Point Types	Digital, Float, Integer (including Real and Imaginary)
Serial-Based interface	Yes

** See paragraphs below for further explanation.*

Uses PI SDK

The PI SDK and the PI API are bundled together and must be installed on each interface node. This interface does not specifically make PI SDK calls.

Source of Timestamps

Timestamps are acquired from the C37.118 data stream.

UniInt-based

UniInt stands for Universal Interface. UniInt is not a separate product or file; it is an OSIsoft-developed template used by developers and is integrated into many interfaces, including this interface. The purpose of UniInt is to keep a consistent feature set and behavior across as many OSIsoft interfaces as possible. It also allows for the very rapid development of new interfaces. In any UniInt-based interface, the interface uses some of the UniInt-supplied configuration parameters and some interface-specific parameters. UniInt is constantly being upgraded with new options and features.

The *PI Universal Interface (UniInt) User Guide* is a supplement to this manual.

Disconnected Start-Up

The PI IEEE C37.118 interface is built with a version of UniInt that supports disconnected start-up. Disconnected start-up is the ability to start the interface without a connection to the PI Data Archive. This functionality is enabled by adding **`/cachemode`** to the list of start-up parameters or by enabling disconnected startup using the ICU. Refer to the *PI Universal Interface (UniInt) User Guide* for more details about UniInt Disconnect startup.

SetDeviceStatus

The health point, having the attribute Exdesc = [UI_DEVSTAT], is used to represent the status of the source device. The following events can be written into this point:

- "1 | Starting | UI #.#.#.#" – The interface is currently starting up.
- "Good" – The interface is properly communicating and reading data from all configured PMUs. A value of "Good" does not mean that all points are receiving good values, but it is a good indication that there are no hardware or network problems.
- "3 | 1 devices(s) in error | REP: 1 LEP 1" – The connection to one or more PMUs has failed.
- "4 | Intf Shutdown" – The interface has been shutdown.

Refer to the *PI Universal Interface (UniInt) User Guide* for more information on how to configure health points.

Failover

- PDC / PMU Data Source Level Failover
    - PDC / PMU Data Source Level Failover maximizes interface data availability on the receiving PI Data Archive(s). It enables the interface to obtain data from one of two PMU/PDC data sources. Each PMU/PDC data source must have identical tag definitions and data streams for interface source points.
    - The interface initiates failover if the active source data becomes stale or is not available due to network issues.
- UniInt Failover Support

UniInt Phase 2 Failover provides support for cold, warm or hot failover configurations. The Phase 2 hot failover results in a *no data loss* solution for bi-directional data transfer between the PI Data Archive and the Data Source given a single point of failure in the system architecture similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition. This failover solution requires that two copies of the interface be installed on different interface nodes, both of which are collecting data simultaneously from a single data source. Phase 2 Failover requires that each interface have access to a shared data file. Failover operation is automatic and operates with no user interaction. Each interface participating in failover has the ability to monitor and determine liveliness and failover status. To assist in administering system operations, the ability to manually trigger failover to a desired interface is also supported by the failover scheme.

The failover scheme is described in detail in the *PI Universal Interface (UniInt) User Guide*, which is a supplement to this manual. Details for configuring this interface to use failover are described in the UniInt Failover Configuration section of this manual.

This interface supports UniInt Failover (Phase 2 Cold and Hot).

Vendor Hardware Required

A C37.118-compliant Phasor Measurement Unit or Phasor Data Concentrator is required for proper communication with this interface.

Device Point Types

The interface supports all Phasor, Analog and Digital channels plus frequency and frequency rate of change. Frequency error is also computed by the interface.

Serial-Based Interface

This interface can run either with a serial connection or Ethernet network connection.

Server class machines often have low-quality serial ports. Server class machines are not required for most interfaces and should not be used, especially when serial port connections are required.

Diagram of Hardware Connection

Chapter 2. Principles of Operation

The PI C37.118 interface is designed to request and process data from C37.118 compliant devices. The PI C37.118 interface is a real-time based interface. The interface operations can be categorized into the following list of sub-operations:

- Startup Process
- Loading PI Points
- Data Processing
- Data Flow to the PI System
- Data Quality Indicators
- Interface Robustness

A more detailed explanation of the list of sub-operations for this interface is provided in this section.

Startup Process

The PI C37.118 interface must be configured in accordance with the Startup Command File (`.bat` file) and the Interface Configuration File (XML configuration file). Each instance of the C37118 interface is assigned a specific identification number and/or point source, which are correlated with the interface-specific startup file parameters **`/id`**, **`/ps`** and **`/configfile`**.

The C37118 interface establishes the initial connection to PI and reconnects to PI in the event that the connection is lost. If the interface is started while the PI Data Archive is down, the interface periodically attempt to establish a connection until the PI Data Archive is up and running.

The interface is designed to make use of a configuration file to set up the internal operation of the interface. The configuration file is written in XML format. A PI Interface Configuration Utility (ICU) plug-in is available for this interface to assist in the configuration of both the startup batch file and the interface (XML) configuration file. The use of this ICU Control (which is highly recommended) is described in the Startup Command File section of this document.

When the interface starts, it parses the command line parameters and searches for the XML configuration file that is specified in the **`/configfile`** command line parameter. If the **`/configfile=<UNCpath>`** is not found, a default file name (`pic37118.xml`), is assumed. The configuration file is then parsed and the interface instantiates the internal structures that are required for communication with the C37.118 PDC and / or PMUs.

A description of the internal structures and the terms used by the interface are described below.

LocalEndPoint

The LocalEndPoint represents the local (interface side) of a physical network or serial connection to a C37.118 device. Each LocalEndPoint should contain the various LocalEndPoint attributes as well as a single RemoteEndPoint leaf node which contains the RemoteEndPoint attributes.

RemoteEndPoint

The RemoteEndPoint object represents the remote (C37.118 device) side of a physical connection. Currently there can be only one RemoteEndPoint configured for each LocalEndPoint.

PMU

The PMU object represents a C37.118 Phasor Data Concentrator (PDC) or Phasor Measurement Unit (PMU). If the RemoteEndPoint is a PDC, there can be multiple PMU objects associated with the RemoteEndPoint.

Loading PI Points

When the PI C37.118 interface starts, it searches the PI Point Database for points that belong to the interface and creates a point list for the interface. The primary method of establishing a mapping between a PI point and a C37.118 device is through configuration of the `Location2`, `Location3`, `Location5`, and `InstrumentTag` PI point attributes. The correlation between the above attributes and the various C37.118 measurements are listed in the PI Point Configuration section of this document.

PI points for this interface have a one-to-one correspondence with electrical measurements that are available from the C37.118 device. There can be only one PI point for receiving a specific measurement from a given C37.118 device.

For example: if a PI point is configured to read the magnitude from a Phasor channel named "Phasor Volt CH-A" (`InstrumentTag=PhasorMag\Phasor Volt CH-A`) from a specific LocalEndPoint, RemoteEndPoint and PMU combination (**`Location2`**, **`Location3`** and **`Location5`**) there can be no other PI points specified with identical attributes.

After the interface loads a PI point, any additional PI points that have the same attributes as a previously loaded PI point are rejected by the interface and the interface logs an error message. Refer to Appendix A. Error and Informational Messages for more details about messages that are associated with loading PI points.

Data Processing

The PI C37.118 interface processes C37.118 formatted data messages as defined in the IEEE C37.118 specification. The input data is parsed to provide the following:

- Real and Imaginary part of the complex number for each Phasor channel if the PDC / PMU is configured to send Phasor measurements in rectangular format.
- Magnitude and Phase Angle for each Phasor channel for each Phasor channel if the PDC / PMU is configured to send Phasor measurements in polar format.
- Analog channel data
- Digital channel data
- System Frequency
- Frequency deviation (calculated by interface)
- Frequency Rate of Change

Note: If the PMU is configured to send data in rectangular format, the interface computes the polar coordinates; if the PMU is configured to send data in polar format the interface computes the rectangular coordinates.

In addition to the data parsed from the C37.118 data, the interface also provides for input of C37.118 configuration data from the CONFIG2 data block. The configuration data is parsed to provide the following:

- Configuration change count from CFGCNT
- Data transmit rate from DATA_RATE
- Nominal frequency (50Hz / 60Hz)

Optional data quality tags can be configured to receive information related to the quality of the C37.118 data. This quality information is extracted from the time quality, leap second status, and STAT fields of the C37.118 data stream. These include the following:

- Data Valid
- PMU Error
- PMU Sync
- Data Sorting
- Time Unlocked Time
- Trigger Reason

The interface can also store geographical information regarding the location of the PMU if defined in the Interface Configuration File. The Latitude and Longitude of the PMU can be defined in decimal form in the configuration file and stored in the PI Data Archive.

Data Flow to the PI System

Data flow from the C37.118 device to the PI System is unsolicited. The following is an explanation of the sequence of events that take place in order to transfer data from the C37.118 device interface into the PI System.

Phasor data from C37.118 is sent to the interface at a rate which is configured in the C37.118 device and is typically 20, 30, 60 or 120 times per second for 60Hz systems and 25, 50, and 100 times per seconds for 50Hz systems.

When the interface communications plug-in issues the start data command (which is issued at startup and after any communications failure), the C37.118 device starts sending data at the configured rate.

The communications plug-in then processes the incoming data stream. Each C37.118 data frame is processed and the size and CRC verified. The C37.118 frame is then queued to the main interface thread.

The main interface thread of the interface parses and processes C37.118 data frames in the incoming data queue and sends the results to the PI System on a real-time basis. This means that device data, after processing, is sent immediately and is available in the PI System.

The illustration below shows a graphical representation of data flow from the C37.118 PDC / PMU device to the PI System.

Multi-Threading

The PI C37.118 interface is multithreaded. Communications with the PDC / PMU is handled in separate threads from the main interface body. Depending on the Communications Plug-In being used, there will be either 1 or 2 communications threads created. If commands and command responses operate on the same communications port as the C37.118 data stream, a single communications thread will handle all communications with the PDC / PMU. If commands and command responses operate on independent ports then 2 communications threads are created. One thread handles commands and command responses and the second is dedicated to receiving the C37.118 data stream. Data from each communications thread is buffered and then serialized through the main interface thread. This must be done because the PI API is not currently thread safe.

Communications Plug-Ins

The PI C37.118 interface uses “Communications Plug-Ins” to handle differences in the application layer communications for various PDCs and PMUs. This is necessary because the IEEE C37.118 specification does not dictate the implementation.

The following describes the plug-ins distributed with the PI C37.118 interface.

PIC37118_Comm001

This plug-in is used for devices that use a single socket via either UDP or TCP for commands, command responses, and measurement data. This plug-in is also used for serial connections.

PIC37118_Comm002

This plug-in is used for devices that use two sockets. Commands and command responses are performed via one socket and measurement data is returned via a separate socket. Typically the device uses TCP for the commands and UDP for the measurement data.

PIC37118_Comm003

This plug-in is used for devices that use a single socket via either UDP or TCP but do not support output commands. Once the device starts it sends data automatically to a predetermined address and port. This is referred to as “Autonomous Mode” on most devices. It expects that the CONFIG2 block is sent periodically along with the measurement data.

Data Quality Indications

The PI C37.118 interface provides multiple methods to monitor the quality of the data being provided by the C37.118 PDC / PMU.

The following types of data quality indications are available. All of these are optional and can be used alone or in conjunction with other data quality indicators.

- Quality Tags – Quality tags are PI tags that are separate from data tags.
- Conditional Data Tag Status – The interface can be configured to apply one of 4 different rules in response to one or more of the data quality flags being set.

Interface Robustness

Where data acquisition is concerned, the C37118 interface is designed to be as robust as possible. If the interface determines that Phasor data has not been received within the time specified by the **`DataTimeout`** parameter in the configuration file, an entry is made to the *pipc.log* file and the interface attempts to restart the C37.118 phasor data stream. Upon receipt of a successful response from the C37.118 device, the interface begins processing data and logs a message to the `pipc.log` file. The interface will retry failed connection attempts until data flow is restored but will not log further connection failure messages until data flow is restored. Refer to Appendix A. Error and Informational Messages for more details about messages that are associated with transmission loss.

The interface is configured to make use of the **`WriteIOTimeout`** parameter in the interface configuration file. This parameter instructs the interface about whether to write an error state to PI points when data from the device is not received after the period specified by the **`DataTimeout`** parameters. Setting **`WriteIOTimeout=1`** (the default), causes the system digital state of **`"I/O Timeout"`** to be written to PI points. Alternately, setting **`WriteIOTimeout=0`** instructs the interface not to update PI points during transmission loss of C37.118 data. The default behavior is to write "I/O Timeout" for all measured values.

When the interface starts, the interface searches the PI Point Database for points that belong to the interface and a point list is created for the interface.

When startup is complete, the interface enters the processing loop, which includes the following:

- Servicing scheduled input points. Each Scan Class is processed in turn.
- Servicing triggered input points as events arrive.
- The PI Point Database is checked every 2 minutes for points that are added, edited, and deleted. If point updates are detected, the points are loaded (or reloaded) by the interface as appropriate. The 2-minute update interval can be adjusted with the **`/updateinterval`** command-line parameter described in the *PI Universal Interface (UniInt) User Guide*. The interface will only process 25 point updates at a time. If more than 25 points are added, edited, or deleted at one time, the interface will process the first 25 points, wait 30 seconds (or by the time specified by the **`/updateinterval`** parameter, whichever is lower), process the next 25 points, and so on. After all points have been processed, the interface resumes checking for updates every 2 minutes (or by the time specified by the **`/updateinterval`** parameter). The interface writes the digital state `SCAN OFF` to any points that are removed from the interface while it is running.

PDC / PMU Data Source Level Failover

PDC / PMU Data Source Level Failover maximizes interface data availability on the receiving C37.118 data stream. It requires that two PDC / PMU data sources are available that have identical Phasor, analog and digital channels and data streams for interface points. This requirement ensures that the interface obtains the same data regardless of which data source is active.

Failover is enabled by specifying a pair of primary and secondary Local End Points by specifying Local End Point and Failover Local End Pont IDs in the xml configuration file. On startup, the interface attempts to establish a connection to the primary data source, then load and initialize its tag list. Data collection begins from the primary PDC / PMU data source making it the active node while the secondary PDC / PMU data source assumes the standby (backup) role.

To set up PDC / PMU Data Source Level Failover you must create at least two Local End Points. You should bind the end points together to create a redundant pair. To do so, use ID and Failover ID fields. Failover ID should be equal to the ID of the redundant Local End Point and vice versa. Only one Local End Point in the redundant pair should be specified as a primary. Use the Primary field to specify which Local End Point is a primary. Another one will be a secondary Local End Point.

To control delay between switches from primary to backup PDC /PMU use the Switch Backup Delay parameter. Default value is 60 seconds.

You can have multiple redundant pairs configured for the interface, or you can set up redundant Local End Points and non-redundant (Failover ID field not assigned) together for the same interface.

Failover

There are two conditions that cause the interface to initiate failover: a communication failure or detection of stale data.

Communication Failure

The first failover condition occurs when the interface loses communication to the active PDC / PMU data source. This might be the result of a temporary network outage, shutdown of the active data source, hardware failure, and so on. The interface cannot identify the cause of the the disconnect failure. A disconnection means that the interface did not receive a response from the active source within the timeout period. The interface may initiate a short wait and then attempt to reconnect to the active source before attempting failover. The data timeout, reconnection rate, and reconnection timeout are all user configurable parameters.

The default reconnection timeout period is 60 seconds, which is set as an interface configuration parameter. If the interface is unable to reconnect to the active PDC / PMU during the reconnection timeout period, the interface attempts to connect to the alternate (primary or backup) PDC / PMU.

Detection of Stale Data

The second failover condition is caused by stale source data. The data received by the PDC / PMU is high frequency. Depending on the brand and configuration of each individual PMU data might arrive at a rate of 10, 20, 30 or 60 Hz. If the interface does not receive data during the data timeout interval (which defaults to 1000 milliseconds) the data is marked as timed out and becomes stale. The interface attempts to reconnect with the active PDC / PMU at the period specified by the reconnection rate interval. After the reconnection timeout interval is expired, the interface attempts to connect to the alternate (backup or primary) PDC /PMU. After each reconnection timeout interval the interface attempts to switch back and forth

between the primary and the backup PDC /PMU until it receives data from the first available PDC / PMU.

UniInt Failover

This interface supports UniInt failover. Refer to the UniInt Failover Configuration section of this document for configuring the interface for failover.

Chapter 3. Installation Checklist

If you are familiar with running PI data collection interface programs, this checklist helps you get the interface running. If you are not familiar with PI interfaces, return to this section after reading the rest of the manual in detail.

This checklist summarizes the steps for installing this interface. You need not perform a given task if you have already done so as part of the installation of another interface. For example, you only have to configure one instance of Buffering for every interface node regardless of how many interfaces run on that node.

The Data Collection Steps below are required. Interface Diagnostics and Advanced Interface Features are optional.

Data Collection Steps

1. Confirm that you are able to use PI SMT to configure the PI Data Archive. You need not run PI SMT on the same computer on which you run this interface.
2. If you are running the interface on an interface node, edit the PI Data Archive's Trust Table to allow the interface to read attributes and point data. If a buffering application is not running on the interface node, the PI trust must allow the interface to write data.
3. Run the installation kit for the PI Interface Configuration Utility (ICU) on the interface node if the ICU will be used to configure the interface. This kit runs the PI SDK installation kit, which installs both the PI API and the PI SDK.
4. Run the installation kit for this interface. This kit also runs the PI SDK installation kit which installs both the PI API and the PI SDK if necessary.
5. If you are running the interface on an interface node, check the computer's time zone properties. An improper time zone configuration can cause the PI Data Archive to reject the data that this interface writes.
6. Run the ICU and configure a new instance of this interface. Essential startup parameters for this interface are:

   Point Source (**/PS=***x*)
   Interface ID (**/ID=***#*)
   PI Data Archive (**/Host=***host:port*)
7. If you will use digital points, define the appropriate digital state sets.

8. Build input tags for this interface. Important point attributes and their purposes are:

   Location1 specifies the interface instance ID.
   Location2 specifies the LocalEndPoint ID.
   Location3 specifies the RemoteEndPoint ID. This attribute always reflects the IDCODE of PDC / PMU contained in word 3 of the C37.118 data stream. If the connected device is a PMU (versus a PDC), Location3 and Location5 will be the same.
   Location4 is not used.
   Location5 specifies the PMU ID. This attribute will always match the IDCODE of the PMU that is the source of the data.
   ExDesc is not used.
   InstrumentTag specifies the specific measurement and type.

   PtSecurity must permit read access for the PI identity, group, or user configured in the PI trust that is used by the interface.

   DataSecurity must permit read access (buffering enabled) or read/write access (unbuffered) for the PI identity, group, or user configured in the PI trust that is used by the interface.

   Security Note: When buffering is configured, the DataSecurity attribute must permit write access for the *buffering application's* PI trust or mapping. DataSecurity write permission for the interface's PI trust is required only when buffering is not configured.

9. Start the interface interactively and confirm its successful connection to the PI Data Archive without buffering. (The DataSecurity attribute for interface points must permit write access for the interface's PI trust.)
10. Confirm that the interface collects data successfully.
11. Stop the interface and configure a buffering application (either Bufserv or PIBufss).
12. Start the buffering application and the interface. Confirm that the interface works together with the buffering application by physically removing the connection between the interface node and the PI Data Archive node. (The DataSecurity attribute for interface points must permit write access for the *buffering application's* PI trust or mapping. The interface's PI trust does not require DataSecurity write permission.)
13. Configure the interface to run as a Windows service. Confirm that the interface runs properly as a service.
14. Restart the interface node and confirm that the interface and the buffering application restart.

Interface Diagnostics

1. Configure scan class performance points.
2. Install the PI Interface for Performance Monitor on the interface node.
3. Configure performance counter points.
4. Configure UniInt health monitoring points
5. Configure the I/O Rate point.

Advanced Interface Features

1. Configure the interface for disconnected startup. Refer to the *PI Universal Interface (UniInt) User Guide* for more details on UniInt disconnected startup.
2. Configure UniInt failover; see the UniInt Failover Configuration chapter in this document for details related to configuring the interface for failover.

Chapter 4. Interface Installation

OSIsoft recommends that interfaces be installed on PI interface nodes instead of directly on the PI Data Archive node. A PI interface node is any node other than the PI Data Archive node where the PI Application Programming Interface (PI API) is installed (see the PI API manual). With this approach, the PI Data Archive need not compete with interfaces for the machine's resources. The primary function of the PI Data Archive is to archive data and to service clients that request data.

After the interface has been installed and tested, Buffering should be enabled on the PI interface node. Buffering refers to either PI API Buffer Server (Bufserv) or the PI Buffer Subsystem (PIBufss). For more information about Buffering see the Buffering for PI Interfaces chapter of this manual.

In most cases, interfaces on PI interface nodes should be installed as automatic services. Services keep running after the user logs off. Automatic services automatically restart when the computer is restarted, which is useful in the event of a power failure.

The guidelines are different if an interface is installed on the PI Data Archive node. In this case, the typical procedure is to install the PI Data Archive as an automatic service and install the interface as an automatic service that depends on the PI Update Manager and PI Network Manager services. This typical scenario assumes that Buffering is not enabled on the PI Data Archive node. Bufserv can be enabled on the PI Data Archive node so that interfaces on the PI Data Archive node do not need to be started and stopped in conjunction with PI, but it is not standard practice to enable buffering on the PI Data Archive node. The PI Buffer Subsystem can also be installed on the PI Data Archive.

Naming Conventions and Requirements

In the installation procedure below, it is assumed that the name of the interface executable is `PIC37118.exe` and that the startup command file is called `PIC37118.bat`.

When Configuring the Interface Manually

It is customary for the user to rename the executable and the startup command file when multiple copies of the interface are run. For example, `PIC371181.exe` and `PIC371181.bat` would typically be used for interface number 1, `PIC371182.exe` and `PIC371182.bat` for interface number 2, and so on. When an interface is run as a service, the executable and the command file must have the same root name because the service looks for its command-line parameters in a file that has the same root name.

Interface Directories

PIHOME Directory Tree

The [PIHOME] directory tree is defined by the PIHOME entry in the pipc.ini configuration file. This pipc.ini file is an ASCII text file, which is located in the %windir% directory.

For 32-bit operating systems, a typical pipc.ini file contains the following lines:

```
[PIPC]
PIHOME=C:\Program Files\PIPC
```

For 64-bit operating systems, a typical pipc.ini file contains the following lines:

```
[PIPC]
PIHOME=C:\Program Files (X86)\PIPC
```

The above lines define the root of the PIHOME directory on the C: drive. The PIHOME directory does not need to be on the C: drive. OSIsoft recommends using the paths shown above as the root PIHOME directory name.

Interface Installation Directory

The interface install kit will automatically install the interface to:

PIHOME\Interfaces\C37118\

PIHOME is defined in the pipc.ini file.

Interface Installation Procedure

The C37118 interface setup program uses the services of the Microsoft Windows Installer. Windows Installer is a standard part of Windows operating systems. To install, run the appropriate installation kit.

```
C37118_#.#.#.#_.exe
```

Silent Installation Procedure

To launch a silent installation, type:

```
Setup.exe -f silent.ini
```

The silent.ini file is included in the interface installation kit. You can make site-specific alterations to the file as needed. See the silent.ini file for further information and descriptions of available arguments.

PI Trust for Interface Authentication

A PI Interface usually runs on an interface node as a Windows service, which is a non-interactive environment. In order for an interface to authenticate itself to a PI Data Archive and obtain the access permissions for proper operation, the PI Data Archive must have a PI trust that matches the connection credentials of the interface. Determine if a suitable

PI trust for the interface exists on the PI Data Archive. If a suitable PI trust does not exist, see the Security chapter for instructions on creating a new PI trust.

Installing Interface as a Windows Service

The preferred way to install the C37118 interface service is to use the PI Interface Configuration Utility; however, you can also create it manually.

Security Note: For improved security, OSIsot recommends running the interface service under a non-administrative account, such as a Windows built-in service virtual account, the built-in Network Service account, or a non-administrative account that you create.

Installing Interface Service with PI Interface Configuration Utility

The PI Interface Configuration Utility provides a user interface for creating, editing, and deleting the interface service:

Service Configuration

Service name

The *Service name* box shows the name of the current interface service. This service name is obtained from the interface executable.

ID

This is the service ID that is used to distinguish among multiple instances of the same interface using the same executable.

Display name

The *Display name* text box shows the current Display Name of the interface service. If there is currently no service for the selected interface, the default Display Name is the service name with a "PI-" prefix. Users may specify a different Display Name. OSIsoft suggests that the prefix "PI-" be appended to the beginning of the interface name to indicate that the service is part of the OSIsoft suite of products.

Log on as

The *Log on as:* options allow the interface service to be configured to use a *default Windows Service* account, a *user-specified Windows* account or the *Local System* account.

Default Windows Service account

Select the ***NT Service\PIC371181*** option to configure the internet service to use a *default Windows Service* account.

This figure shows the option to select to configure the interface service to use a *default Windows Service* account.

User-specified Windows account

Select the ***[Domain\]UserName*** option to add a ***user-specified Windows*** account which the interface service will use. Please note that this account differs from the *default Windows Service* account.

This figure shows the option to select to configure the interface service to use a *user-specifed Windows* account.

When the ***[Domain\]UserName*** option is selected, three additional boxes are displayed. These additional boxes are used to enter information about the ***user-defined Windows*** account.

This figure shows an example of how to populate the configuration boxes to allow the interface service to use a *user-specifed Windows* account.

- UserName

  The *UserName:* box allows the entry of the Windows User account to be used by the interface service.

- Password

  If a Windows User account is entered into the *UserName:* box, the password for the entered account must be provided in the *Password:* box, unless the account requires no password.

- Confirm password

  If a password is entered into the *Password:* box, it must be confirmed in the *Confirm Password:* box.

Local System account

Select the ***[Domain\]UserName*** option and populate the *UserName* box with the value *LocalSystem* to configure the interface service as a *Local System* account. Leave the *Password:* and *Confirm password:* boxes empty.

This figure shows an example of how to populate the configuration boxes to configure the interface service to run as a *Local System* account.

After the interface service has been created to use the *LocalSystem* account, the ***Service Configuration / Log on as:*** area is redisplayed with a different dialog view.

This figure shows how the *Service Configuration / Log on as:* area is redrawn when the interface service is configured to use the *Local System* account.

This figure shows the interface after it has been created to use the *Local System* account.

Security Note: For best security, we recommend running this interface service under an account with minimum privileges, such as a Windows built-in service virtual account, the built-in Network Service account, or a non administrative account that you create.

PI ICU versions earlier than 1.4.14.x cannot create a service that runs as a Windows built-in service virtual account or the built-in Network Service or Local Service accounts. After ICU creates the interface service, you can change the account with a Windows administrative tool, such as *Services on the Control Panel* or the *sc* command-line utility.

Dependencies

The *Installed services* list is a list of the services currently installed on this machine.

Services upon which this interface is dependent should be moved into the *Dependencies:* list using the ***Add*** button, . For example, if API Buffering is enabled, then *Bufserv* or *PIBufss* should be selected from the list at the right and added to the list on the left. To remove a service from the list of dependencies, use the ***Remove*** button, , and the service name will be removed from the *Dependencies:* list.

When the interface is started (as a service), the services listed in the dependency list will be verified as running (or an attempt will be made to start them). If the dependent service(s) cannot be started for any reason, then the interface service will not run.

Note: Please see the PI Log and Windows Event Logger for messages that may indicate the cause for any service not running as expected.

- *Add* Button

To add a dependency from the list of *Installed services:*, select the dependency name and click the *Add* button.

- *Remove* Button

To remove a dependency from the list of *Dependencies:*, highlight the service name in the *Dependencies* list and click the *Remove* button.

The full name of the service selected in the *Installed services:* list is displayed below the *Installed services:* list box.

Startup Type

The *Startup Type* indicates whether the interface service will start automatically or must be started manually upon reboot.

- If the *Auto* option is selected, the service will be installed to start automatically when the machine reboots.
- If the *Manual* option is selected, the interface service will not start on reboot, but will require manual intervention to start the service.

- If the *Disabled* option is selected, the service will not start at all.

Generally, interface services are set to start automatically.

Create

The *Create* button adds the displayed service with the specified *Dependencies* and with the specified *Startup Type*.

Remove

The *Remove* button removes the displayed service. If the service is not currently installed, or if the service is currently running, this button will be grayed out.

Start or Stop Service

The toolbar contains a *Start* button ▶ and a *Stop* button ■. If this interface service is not currently installed, these buttons will remain grayed out until the service is added. If this interface service is running, the *Stop* button is available. If this service is not running, the *Start* button is available.

The status of the interface service is indicated in the lower portion of the PI ICU dialog.

Installing Interface Service Manually

Help for installing the interface as a service is available at any time with the command:

```
PIC37118.exe /help
```

Open a Windows command prompt window and change to the directory where the `PIC371181.exe` executable is located. Then, consult the following table to determine the appropriate service installation command.

Note: In the following Windows Service Installation Commands you may use either a slash (/) or dash (-) as the delimiter.

Windows Service Installation Commands on an Interface Node or a PI Data Archive Node with Bufserv implemented	
Manual service	PIC37118.exe /install /depend “tcpip bufserv”
Automatic service	PIC37118.exe /install /auto /depend “tcpip bufserv”
*Automatic service with service ID	PIC37118.exe /serviceid *X* /install /auto /depend “tcpip bufserv”
Windows Service Installation Commands on an Interface Node or a PI Data Archive Node without Bufserv implemented	
Manual service	PIC37118.exe /install /depend tcpip
Automatic service	PIC37118.exe /install /auto /depend tcpip
*Automatic service with service ID	PIC37118.exe /serviceid *X* /install /auto /depend tcpip

*When specifying service ID, the user must include an ID number. It is suggested that this number correspond to the interface ID (**`/id`**) parameter found in the interface .bat file.

Check the Microsoft Windows Services control panel to verify that the service was added successfully. The services control panel can be used at any time to change the interface from an automatic service to a manual service or vice versa.

The service installation commands in this section create an interface service that runs under a low-privilege built-in account. On Windows 7 and Server 2012 and later, the service logs on as the service virtual account. For earlier versions of Windows, the service logs on as Network Service.

Security Note: For best security, we recommend running this interface service under an account with minimum privileges, such as a Windows service virtual account, the built-in Network Service account, or a non administrative account that you create.

As discussed earlier, following the recommendation to run the interface service under a low-privilege account may affect performance counters.

The services control panel can change the account that the interface service runs under. Changing the account while the interface service is running does not take effect until the interface service is restarted.

Chapter 5. Digital States

For more information regarding Digital States, refer to the PI Data Archive documentation.

Digital State Sets

PI digital states are discrete values represented by strings. These strings are organized in the PI Data Archive as digital state sets. Each digital state set is a user-defined list of strings, enumerated from 0 to n to represent different values of discrete data. For more information about PI digital tags and editing digital state sets, see the PI Data Archive manuals.

An interface point that contains discrete data can be stored in the PI Data Archive as a digital point. A digital point associates discrete data with a digital state set, as specified by the user.

Interface Specific Digital Sets

If quality tags are being used, quality point specific digital sets must be created. Example structure files are provided to assist in the configuration. Configuration files for the following digital state sets are provided.

- PIC37118_CompositeQual
- PIC37118_ConfigChange
- PIC37118_ConnectState
- PIC37118_DataSorting
- PIC37118_DataVaild
- PIC37118_LeapSecond
- PIC37118_NominalFreq
- PIC37118_PMUError
- PIC37118_SyncError
- PIC37118_Timelock
- PIC37118_TimeQuality
- PIC37118_Trigger

The example structure files provided can be used to import the sets into PI using SMT. They are located in the `Configuration Files` subdirectory of the specified installation location.

System Digital State Set

Similar to digital state sets is the system digital state set. This set is used for all points, regardless of type, to indicate the state of a point at a particular time. For example, if the interface receives bad data from the data source, it writes the system digital state `Bad Input` to the PI point instead of a value. The system digital state set has many unused states that can be used by the interface and other PI clients. Digital States 193-320 are reserved for OSIsoft applications.

Chapter 6. PointSource

The PointSource is a unique, single or multi-character string that is used to identify the PI point as a point that belongs to a particular interface. For example, the string *Boiler1* may be used to identify points that belong to the *MyInt* interface. To implement this, the PointSource attribute would be set to Boiler1 for every PI point that is configured for the *MyInt* interface. Then, if **/ps=Boiler1** is used on the startup command-line of the *MyInt* interface, the interface will search the PI Point Database upon startup for every PI point that is configured with a PointSource of Boiler1. Before an interface loads a point, the interface usually performs further checks by examining additional PI point attributes to determine whether a particular point is valid for the interface. For additional information, see the **/ps** parameter. If the PI API version being used is earlier than 1.6.x or the PI Data Archive version is earlier than 3.4.370.x, the PointSource is limited to a single character unless the SDK is being used.

Case-sensitivity for PointSource Attribute

The PointSource character that is supplied with the **`/ps`** command-line parameter is not case sensitive. That is, **`/ps=P`** and **`/ps=p`** are equivalent.

Reserved Point Sources

Several subsystems and applications that ship with the PI System are associated with default PointSource characters. The Totalizer Subsystem uses the PointSource character `T`, the Alarm Subsystem uses @ for Alarm points, `G` for Group Alarms and `Q` for SQC Alarm points, Random uses `R`, RampSoak uses `9`, and the Performance Equations Subsystem uses `C`. Do not use these PointSource characters or change the default point source characters for these applications. Also, if a PointSource character is not explicitly defined when creating a PI point; the point is assigned a default PointSource character of `Lab` (PI 3). Therefore, it would be confusing to use `Lab` as the PointSource character for an interface.

Note: Do not use a point source character that is already associated with another interface program. However it is acceptable to use the same point source for multiple instances of an interface.

Chapter 7. PI Point Configuration

The PI point is the basic building block for controlling data flow to and from the PI Data Archive. A single point is configured for each measurement value that needs to be archived.

Point Attributes

Use the point attributes below to define the PI point configuration for the interface, including specifically what data to transfer.

This document does not discuss the attributes that configure UniInt or PI Data Archive processing for a PI point. Specifically, UniInt provides exception reporting and the PI Data Archive or PIBufss provides data compression. Exception reporting and compression are very important aspects of data collection and archiving, which are not discussed in this document.

Note: See the *PI Universal Interface (UniInt) User Guide* and PI Data Archive documentation for information on other attributes that are significant to PI point data collection and archiving.

Tag

The Tag attribute (or tag name) is the name for a point. There is a one-to-one correspondence between the name of a point and the point itself. Because of this relationship, PI System documentation uses the terms “tag” and “point” interchangeably.

Follow these rules for naming PI points:

- The name must be unique in the PI Data Archive.
- The first character must be alphanumeric, the underscore (_) or the percent sign (%).
- Control characters such as linefeeds or tabs are illegal.
- The following characters also are illegal: * ’ ? ; { } [ ] | \ ` ‘ “

Length

Depending on the version of the PI API and the PI Data Archive, this interface supports Tag attributes whose length is at most 255 or 1023 characters. The following table indicates the maximum length of this attribute for all the different combinations of PI API and PI Data Archive versions.

PI API	PI Data Archive	Maximum Length
1.6.0.2 or later	3.4.370.x or later	1023
1.6.0.2 or later	Earlier than 3.4.370.x	255
Earlier than 1.6.0.2	3.4.370.x or later	255
Earlier than 1.6.0.2	Earlier than 3.4.370.x	255

If the PI Data Archive version is earlier than 3.4.370.x or the PI API version is earlier than 1.6.0.2, and you want to use a maximum Tag attribute length of 1023, you need to enable the PI SDK. See Appendix B. PI SDK Options for information.

PointSource

The PointSource attribute contains a unique, single or multi-character string that is used to identify the PI point as a point that belongs to a particular interface. For additional information, see the **`/ps`** command-line parameter and the PointSource chapter.

PointType

Typically, device point types do not need to correspond to PI point types. For example, integer values from a device can be sent to floating point or digital PI tags. Similarly, a floating-point value from the device can be sent to integer or digital PI tags, although the values will be truncated.

Float16, float32, float 64, int16, int32, digital, string, and blob point types are supported. For more information on the individual point types, see the PI Data Archive manuals.

Location1

Location1 indicates to which copy of the interface the point belongs. The value of this attribute must match the **`/id`** command-line parameter.

Location2

Location2 specifies the ID of the LocalEndPoint. This corresponds to the `ID` parameter of the corresponding LocalEndPoint section in the interfaces XML configuration file. This attribute is used by the interface to uniquely identify the LocalEndPoint object and does not correspond to any C37.118 attributes.

Note: The ID Must Be Unique among all LocalEndPoint sections in the configuration file.

Location3

Location3 specifies the ID of the RemoteEndPoint. This corresponds to the IdCode attribute of the corresponding RemoteEndPoint section in the interfaces XML configuration file and must match the configured IDCODE on the PDC / PMU hardware. If the C37.118 device is a PDC there will most likely be multiple PMUs. This attribute should match the IDCODE field for the PDC and not the PMU.

Location4

The C37.118 interface is exception based only. Location 4 should be set to zero for all points.

Location5

Location5 specifies the PMU ID. This attribute must match the IDCODE field for the PMU in the C37.118 CONFIG2 message as well as the IdCode attribute in the PMU section of the XML configuration file. The value for this will be determined when configuring PDC or PMU. If the C37.118 device is a PMU, versus a PDC, this value will typically match the IDCODE for the RemoteEndPoint specified in location3.

If the device is a PDC, each PMU being collected by the PDC must have a unique IDCODE field configured.

InstrumentTag

The InstrumentTag attribute specifies the specific C37.118 measurement to be stored in the PI Data Archive. The attribute contains two elements. The first is the Measurement Type and the second is the specific channel name, REP or PMU measurement and is in the format of **MeasurementType\Measurement**.

Length

Depending on the version of the PI API and the PI Data Archive, this interface supports an `InstrumentTag` attribute whose length is at most 32 or 1023 characters. The following table indicates the maximum length of this attribute for all the different combinations of PI API and PI Data Archive versions.

PI API	PI Data Archive	Maximum Length
1.6.0.2 or later	3.4.370.x or later	1023
1.6.0.2 or later	Earlier than 3.4.370.x	32
Earlier than 1.6.0.2	3.4.370.x or later	32
Earlier than 1.6.0.2	Earlier than 3.4.370.x	32

If the PI Data Archive version is earlier than 3.4.370.x or the PI API version is earlier than 1.6.0.2, and you want to use a maximum InstrumentTag length of 1023, you need to enable the PI SDK. See Appendix B. PI SDK Options for information.

Measurement Types

There are five primary Measurement Types that the interface supports. They are:

- Phasor – The interface will read one of the C37.118 Phasor channels. There are four subtypes for Phasor measurements.
- Analog – The interface will read one of the C37.118 Analog channels
- Digital – The interface will read one of the C37.118 Digital channels.
- PMU – PMU measurement types are those which are not Phasor, Analog or Digital and are related to the PMU. These include PMU values such Frequency (FREQ) and Frequency rate of Change (DFREQ), Configuration data such as the data scheduling time base (TIME_BASE) field from the C37.118 CONFIG message and interface configuration data from the XML configuration file such as the PMU Latitude and Longitude.
- REP – REP measurement types are those related to the RemoteEndPoint. These include connection status as well as time quality.

For Phasor measurement types, the interface can return four different measurements: Real, Imaginary, Magnitude and Phase Angle.

The following table illustrates the valid MeasurementType keywords for use in the InstrumentTag attribute. The keywords are not case sensitive.

Keyword	Description
PhasorReal	Returns the Real measurement for the specified Phasor Channel.
PhasorImag	Returns the Imaginary measurement for the specified Phasor Channel.
PhasorMag	Returns the Magnitude measurement for the specified Phasor Channel.
PhasorAngle	Returns the Phase Angle measurement for the specified Phasor Channel
Analog	Returns the measurement from the specified Analog Channel.
Digital	Returns the Digital measurement from the specified Digital Channel.
PMU	The interface returns one of the PMU measurements.
REP	Remote End Point related tag

Measurements

For Phasor, Analog and Digital Measurement types, the user must specify the C37.118 channel name from which the interface will collect data. Channel names vary by vendor.

For the PMU and REP Measurement types, the user must enter one of the PMU or REP Measurement keyword described below.

The C37.118 specification defines the channel names to be a 16-byte ASCII string that can have embedded as well as trailing spaces. The interface strips any trailing spaces but leaves any embedded spaces. The interface also treats the strings as case insensitive. If for example a PMU has a channel name of ‘Watts CH-A ‘, the user would specify ‘Watts CH-A’ for the Measurement element of the InstrumentTag (‘watts ch-a’ would also be valid in this example).

For PMU MeasurementTypes tags, the interface will store various data from the C37.118 data stream, CONFIG1 and CONFIG2 blocks and configuration data from the XML configuration file. The follow table shows the valid values for the Measurement field when the MeasurmentType field is PMU.

Keyword	Description
FREQ	System frequency (FREQ) from C37.118 data stream
DFREQ	Frequency Rate of Change (DFREQ) from C37.118 data stream.
FREQERR	Frequency error calculated from FREQ – Nominal Frequency as defined by FNOM (50 or 60).
CFGCNT	Configuration change count (CFGCNT) from last CONFIG block
CONFIGFLAG	Digital tag that is set to 1 when the configuration change flag (bit 10) in the STAT word is true.
FNOM	Nominal line frequency (FNOM) from last CONFIG block
DATA_RATE	Date rate (DATA_RATE) from last CONFIG block
LATITUDE	PMU Latitude (-90.0-90.0) from XML configuration file
LONGITUDE	PMU Longitude (-180.0-180.0) from XML configuration file
TIMELOCK	Digital tag receives the Unlocked Time quality bits (5-4) from the STAT word. A sample digital set definition file **`PIC37118_Timelock.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
TRIGGER	Digital tag receives the Trigger Reason bits (3-0) from the STAT word. A sample digital set definition file **`PIC37118_Trigger.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
DATAVALID	Digital tag receives the Data Validity bit (15) from the STAT word. A sample digital set definition file **`PIC37118_DataValid.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
PMUERROR	Digital tag receives the PMU Error bit (14) from the STAT word. A sample digital set definition file **`PIC37118_PMUError.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
SYNCERROR	Digital tag receives the PMU Sync Error bit (13) from the STAT word. A sample digital set definition file **`PIC37118_SyncError.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
DATASORTING	Digital tag receives the Data Sorting bit (12) from the STAT word. A sample digital set definition file **`PIC37118_DataSorting.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
COMPOSITEQUAL	Digital tag receives a composite status from the Data Valid, PMU Error, Sync Error and Data Sorting flags. If any of these flags are set, the value of this tag will be set to 1. Otherwise it will be zero. This tag is intended to be used to enable the user to make a single check for any quality related issues and then do further checking only if needed. A sample digital set definition file **`PIC37118_CompositeQuality.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.

The following table lists measurement values for PMU measurement type tags.

Keyword	Description
TIMEQUAL	Digital tag receives the Qime Quality bits (3-0) from the time quality flags. A sample digital set definition file **`PIC37118_TimeQuality.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
LEAPSEC	Digital tag receives the Leap Second quality bits (6-4) from the time quality flags. A sample digital set definition file **`PIC37118_LeapSecond.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
CONNECTSTATE	Digital tag receives the current connection state. A sample digital set definition file **`PIC37118_ConnectState.csv`** is provided with the interface to assist in creating the required digital state set for this tag. This file can be used to import the state set into PI using the Digital States Plug-In in SMT.
EVENTSEQ	Integer tag intended for diagnostic purposes to verify that the interface is not losing messages. It records the sequence number of the message within a second. For example: If the update rate for the PMU is set to 30hz, the expected values would cycle from 0 to 29 without missing any values.

ExDesc

This attribute is not used by the PI C37.118 interface.

Scan

By default, the Scan attribute has a value of 1, which means that scanning is turned on for the point. Setting the scan attribute to 0 turns scanning off. If the scan attribute is 0 when the interface starts, a message is written to the log and the point is not loaded by the interface. There is one exception to the previous statement:

If any PI point is removed from the interface while the interface is running (including setting the scan attribute to 0), `SCAN OFF` will be written to the PI point regardless of the value of the Scan attribute. Two examples of actions that would remove a PI point from an interface are to change the point source or set the scan attribute to 0. If an interface-specific attribute is changed that causes the point to be rejected by the interface, `SCAN OFF` will be written to the PI point.

Shutdown

The Shutdown attribute is 1 (true) by default. The default behavior of the PI Shutdown subsystem is to write the `SHUTDOWN` digital state to all PI points when PI is started. The timestamp that is used for the `SHUTDOWN` events is retrieved from a file that is updated by the Snapshot Subsystem. The timestamp is usually updated every 15 minutes, which means that the timestamp for the `SHUTDOWN` events will be accurate to within 15 minutes in the event of a power failure. For additional information on shutdown events, refer to PI Data Archive manuals.

Note: The `SHUTDOWN` events that are written by the PI Shutdown subsystem are independent of the `SHUTDOWN` events that are written by the interface when the **`/stopstat=Shutdown`** command-line parameter is specified.

`SHUTDOWN` events can be disabled from being written to PI points when the PI Data Archive is restarted by setting the Shutdown attribute to 0 for each point. Alternatively, the default behavior of the PI Shutdown Subsystem can be changed to write `SHUTDOWN` events only for PI points that have their Shutdown attribute set to 0. To change the default behavior, edit the `\PI\dat\Shutdown.dat` file, as discussed in PI Data Archive manuals.

Bufserv and PIBufss

It is not recommended to write shutdown events when buffering is being used. Bufserv and PIBufss are utility programs that provide the capability to store and forward events to a PI Data Archive, allowing continuous data collection when the PI Data Archive is down for maintenance, upgrades, backups, or unexpected failures. That is, when the PI Data Archive is shut down, Bufserv or PIBufss will continue to collect data for the interface, making it undesirable to write `SHUTDOWN` events to the PI points for this interface. Disabling Shutdown is recommended when sending data to a Highly Available PI Data Archive Collective. Refer to the Bufserv or PIBufss manuals for additional information.

DataSecurity

The PI identity in the PI trust that authenticates the interface must be granted read access by the DataSecurity attribute of every PI point that the interface services. If the interface is used *without* a buffering application, write access also must be granted. (If the interface is used with a buffering application, the buffering application requires write access but the interface does not.)

PtSecurity

The PI identity in the PI trust that authenticates the interface must be granted read access by the PtSecurity attribute of every PI point that the interface services.

Chapter 8. Startup Command File

Command-line parameters can begin with a `/` or with a `-`. For example, the **`/ps=M`** and **`-ps=M`** command-line parameters are equivalent.

Command file names have a .bat extension. The Windows continuation character (`^`) allows for the use of multiple lines for the startup command. The maximum length of each line is 1024 characters (1 kilobyte). The number of parameters is unlimited, and the maximum length of each parameter is 1024 characters.

The PI Interface Configuration Utility (PI ICU) provides a tool for configuring the interface startup command file.

Configuring the Interface with PI ICU

Note: PI ICU requires PI 3.3 or later.

The PI Interface Configuration Utility provides a graphical user interface for configuring PI interfaces. If the interface is configured by the PI ICU, the batch file of the interface (`PIC37118.bat`) will be maintained by the PI ICU and all configuration changes will be kept in that file and the module database. The procedure below describes the necessary steps for using PI ICU to configure the C37118 interface.

From the PI ICU menu, select *Interface*, then *NewWindows Interface Instance from EXE...*, and then *Browse* to the `PIC37118.exe` executable file. Then, enter values for *Host PI System*, *Point Source* and *Interface ID#*. A window such as the following results:

Interface name as displayed in the ICU (optional) will have PI- pre-pended to this name and it will be the display name in the services menu.

Click *Add*.

The following message displays:

Note that in this example the host PI Data server is JMURASKI7440. To configure the interface to communicate with a remote PI Data server, select *Connections…* on the PI ICU *Interface* menu and select the default server. If the remote node is not present in the list of servers, it can be added.

After the interface is added to PI ICU, near the top of the main PI ICU screen, the Interface *Type* should be `C37118`. If not, use the drop-down box to change the Interface *Type* to be `C37118`."

Click *Apply* to enable the PI ICU to manage this copy of the C37118 interface.

The next step is to make selections in the interface-specific tab (i.e. "`C37118`") that allow the user to enter values for the startup parameters that are particular to the C37118 interface.

Since the C37118 interface is a UniInt-based interface, in some cases the user will need to make appropriate selections in the *UniInt* page. This page allows the user to access UniInt features through the PI ICU and to make changes to the behavior of the interface.

To set up the interface as a Windows Service, use the *Service* page. This page allows configuration of the interface to run as a service as well as to starting and stopping of the interface service. The interface can also be run interactively from the PI ICU. To do that, select *Start Interactive* on the *Interface* menu.

For more detailed information on how to use the above-mentioned and other PI ICU pages and selections, please refer to the *PI Interface Configuration Utility* user guide. The next section describes the selections that are available from the *C37118* page. Once selections have been made on the PI ICU GUI, press the *Apply* button in order for PI ICU to make these changes to the interface's startup file.

C37118 Interface page

Since the startup file of the C37118 interface is maintained automatically by the PI ICU, use the *C37118* page to configure the startup parameters; you should avoid making changes to the file manually. The following is the description of interface configuration parameters that are used in the PI ICU Control and the corresponding manual parameters.

C37118

The PI Interface for IEEE C37.118 - ICU has one tab. A yellow text box indicates that an invalid value has been entered, or that a required value has not been entered.

Path to XML Config File

This box is used to specify the path and filename of the XML Configuration file to be read by the interface. If the path contains spaces, then the path and filename should be enclosed in quotation marks. To enter a <UNC Path> click the Browse button and choose an existing file from the resulting dialog box (shown below) or enter a new filename in the File name: text box at the bottom of the screen **(/CONFIGFILE=***<UNC Path>***)**

Additional Parameters

This section is provided for any additional parameters that the current ICU Control does not support.

XML Parameters

Session

The session section contains configuration parameters that control the behavior of the interface when various error conditions occur.

The following figure illustrates the Session section of the C37.118 ICU.

- Sync Error Action – Controls the behavior of the interface when the SYNC flag is set in the C37.118 data stream. (`SyncErrorAction=#`, Default: 1 (Mark as Questionable), Range 0-3)
- Data Sorting Action – Controls the behavior of the interface when the Data Sorting flag is set in the C37.118 data stream. This flag is set on in a PDC data stream when a PMU loses time synchronization making it impossible for the PDC to time correlate the Phasor data. (`DataSortingAction=#`, Default: 1 (Mark as Questionable), Range 0-3)
- Data Valid Action – Controls the behavior of the interface when the Data Valid flag is set in the C37.118 data stream. This flag is set in a PDC data stream when the PDC loses connection to a PMU. When this happens, the PDC typically zero fills the data. (`DataValidAction=#`, Default: 3 (Disard the Data), Range 0-3)
- PMU Error Action – Controls the behavior of the interface when the PMU Error flag is set in the C37.118 data stream. This flag is set when an error occurs on a PMU and indicates a possible error in the data. (`PMUErrorAction=#`, Default: 3 (Discard the Data), Range 0-3)
- Leap Second Action – Controls the behavior of the interface when the Leap Second flag is set in the C37.118 data stream. This flag indicates that a leap second has occurred. (`LeapSecondAction=#`, Default: 0 (Store the Value), Range 0-3)
- Write IO Timeout – If set to yes, the interface will write the system digital state of I/O Timeout when no C37.118 data messages are received within the timeframe specified in the LocalEndPoint section. (`WriteIOTimeout=#`, Default: 1 (Yes), Range:0-1)
- Skip CONFIG2 Command – If set to yes, the interface will send CONFIG2 command to PDC /PMU only during startup. Otherwise, the interface will send CONFIG2 request each time when it is trying to reconnect to PDC / PMU. (`SkipConfig2Cmd=#`, Default: 0 (No), Range:0-1)
- Flush Events – If set to yes, the interface will flush all data to the PI Data Archive after each C37.118 data messages received from PDC /PMU. Otherwise, it will flush

data when buffer gets full (192 records) (`FlushEvents=#`, Default: 0 (No), Range:0-1).

- Send Stop Data Command – If set to yes, the interface will send DATA_STOPCOMMAND to the PDC / PMU when user shutdown the interface (`SendStopCmd=#`, Default: 0 (NO), Range:0-1)
- Switch To Backup Delay – Controls the behavior of the interface for the PDC /PMU Data Source Level Failover. If the interface is not able to set connection and receive data from primary PDC / PMU during this period of time it will switch and try to connect to the backup PDC / PMU. If it loses connection to backup and is not able to reconnect to backup PDC / PMU again during this period of time it will try to reconnect to the primary PDC / PMU again. (`SwitchToBackupDelay=#`, Default: 60 sec (Store the Value), Range 30-600)

The available selections for Sync Error, Data Sorting, Data Valid, PMU Error and Leap Second Actions above are as follows.

- 0 (Store the Value) – The interface will store the value normally.
- 1 (Mark as Questionable) – The interface will store the value but mark it as questionable.
- 2 (Store the Corresponding System Digital State) – The interface will store the corresponding system state.
- 3 (Discard the Data) – The interface will not store the value in the PI Data Archive.

LocalEndPoint

The LocalEndPoint section defines parameters specific to the interface side of a PDC / PMU connection.

The following illustrates the format of the LocalEndPoint section of the C37.118 ICU.

- IP – Identifies this LocalEndPoint as using a TCP connection (`Type=<string>`, Range: IP or Serial)
- Serial – Identifies this LocalEndPoint as using a Serial connection (`Type=<string>`, Range: IP or Serial)
- ID – Specifies the numeric ID of the LocalEndPoint. The ID uniquely identifies the LocalEndPoint when configuring tags for an interface. The ID must be unique among all LocalEndPoints. (`Id=#`, Default:None, Range: 0-65535, Required)
- Comm Dll – Defines the Communications Plug-In to be used to communicate with the PDC / PMU. Due to the variation in application level functionality from one device vendor to another, the plug-in approach allows for new devices to be easily supported. Refer to Appendix C. Communication Plug-In Selection in this manual or to the OSIsoft support site for details on available plug-in's.
- Data Timeout – Controls the length of time to wait, in milliseconds, for data responses. The recommended value is 5 times the configured PMU data rate. (`DataTimeout=#`, Default:1000, Range: 10-65,535)
- Enabled – Allows the LocalEndPoint and everything associated with it to be disabled and not loaded by the interface. It can then be re-enabled with having to reconfigure. (`Enabled=#`, Default: 1 (Yes), Range: 0-1)
- Reconnect Rate – Controls the rate in which the interface attempts to re-establish communications with the PDC / PMU. (`ReconnectRate=#`, Default:30000, Range: 10-65,535)
- Primary – Allows to specify which LocalEndPoint in the redundant pair becomes a primary (value should be 1). For another LocalEndPoint this field should have opposite value (0) to become a backup. Use this field to setup Data Source Level Failover. (`Primary=#`, Default: 0 (No), Range: 0-1)
- Failover ID – Specifies the numeric Failover ID of the LocalEndPoint. The Failover ID allows creating redundant pair of primary and secondary LocalEndPoints for Data Source Level Failover. For example, if ID for LocalEndPoint is 1and Failover ID is 2 then another for LocalEndPoint with ID 2 and Failover ID 1 has to be created. In this case LocalEndPoint 1 and 2 becomes redundant pair. You should specify which

LocalEndPoint is primary using Primary field. Only one LocalEndPoint in the redundant pair could be a primary. Another one becomes a secondary. If no need to setup Data Source Level Failover for this LocalEndPoint leave this field blank. Use this field only in case to setup Data Source Level Failover. (`FailOverId=#`, Default:None, Range: 0-65535, Optional)

IP Configuration

The following controls are available when Type="IP" is selected.

- Command Protocol – Specifies the protocol to use for C37.118 Command and Configuration messages. (`CommandProtocol=<type>`, Default: TCP, Range: TCP or UDP)
- Local Command Port – Specifies the local port to listen on for configuration packets. If not specified the interface will not specify a specific port (system assigned). This parameter will typically be used when the command protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the Local Data Port value. (`LocalCommandPort=#`, Default: 0 (System Assigned), Range: 1024-65,535)
- Data Protocol – Specifies the protocol to use for C37.118 data messages. (`DataProtocol=<type>`, Default: UDP, Range: TCP or UDP)
- Local Data Port – Specifies the local port to listen on for data packets. If not specified the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. For some PDC / PMU configurations in which the device is sends to a predefined data port, this value will also need to be specified. It is also required if Multicast being used. (`LocalDataPort=#`, Default: 0 (System Assigned), Range: 1024-65,535)
- Close Command Socket – If Yes is specified, the interface will close the command socket (if different than the data socket) once a command is sent and the expected response is successfully received. Some devices, such as the Arbiter 1133, expect that the command socket be closed after use. Others, such as SEL devices require the socket remain open. (`CloseCommandSocket=#`, Default: 1 (Yes), Range: 0 (No) – 1 (Yes))
- Use Local IP Address – This parameter could be used if interface node has multiple IP addresses. When specified, the interface will be binding with that address. Otherwise the interface will be using default IP address. (`LocalIPAddress=<ipaddr>`, Default: None)
- Use Multicast IP Addr – This parameter is valid only when the Local Data Protocol is specified as UDP. When specified, the interface will be configured to listen for C37.118 data messages via Multicast. (`MulticastIPAddress=<ipaddr>`, Default: None)

- Src Specific Multicast – This parameter is valid only if the Local IP Address and Multicast IP Address fields have been populated. When specified, the interface will subscribe for multicast data from the source that has specified IP address. (**`MulticastSourceIPAddress=`**`<ipaddr>`, Default: None)

Serial Configuration

The following controls are available when Type="Serial" is selected.

- Baud – Specifies speed at which to communicate with the PDC / PMU. Refer to your PDC / PMU hardware users guide for supported baud rates. (`Baud=#`, Default: 38400, Range: 300-115200)
- COM Port – The numeric value of the serial COM port to use. For example if using COM1 set this value to 1. (`Port=#`, Range: 1-Max Ports)
- Parity – Specifies the error checking parity to use. (`Parity=#`, Default: 0 (NONE), Values: 0 (NONE), 1 (ODD), 2 (EVEN), 3 (MARK), and 4 (SPACE))
- Stop Bits – Specifies the number of stop bits following a message. (`StopBits=#`, Default: 0 (1 bit), Values: 0 (1 bit), 1 (2 bits), 2 (1.5 bits))

RemoteEndPoint

The RemoteEndPoint represents the remote (PDC / PMU side) of a physical connection. Each RemoteEndPoint node will have one or more PMU child nodes. Each of the PMU nodes will represent a PMU that the PI C37.118 interface will communicate with through the physical port described by the LocalEndPoint / RemoteEndPoint child node pair.

RemoteEndPoint Child Nodes that apply to all connection types.

- ID Code – Specifies the numeric ID of the PDC or PMU. This must match the IDCODE field sent by the C37.118 device. If the device is a PDC it is important to note that this must match the IDCODE of the PDC and not the PMU(s) being serviced by the PDC. (`IdCode=#`, Range: 0-32,767)
- Description – User defined description of the RemoteEndPoint. Ie. "Substation xyz". This attribute is used by the interface when logging messages regarding the RemoteEndPoint. (`Description=`*`<string>`*)

The following illustrates the RemoteEndPoint section for an IP LocalEndPoint.

- Remote Data Port – Specifies the remote (PDC) port to be used for data packets. If not specified the interface will assume a specific port (system assigned). If this field is specified, the data being received from the PDC / PMU must originate from this port. Most PDCs and PMUs use a random system assigned port. (`RemoteDataPort=#`, Default: 4712, Range: 1024-65,535)
- Remote Command Port – Specifies the remote port to use for command and configuration packets. The interface will send CONFIG1, CONFIG2 and HEADER request commands to this port. (`RemoteCommandPort=#`, Default: 4713, Range: 1024-65,535)
- Host Name – If this control is selected the PDC / PMU addressing information is passed by host name.(`HostName=<nodename>` or `<IPAddr>`)
- IP Address – If this control is selected the PDC / PMU addressing information is passed as an IP address in dotted decimal notation. (`HostName=<nodename>` or `<IPAddr>`)

The following illustrates the RemoteEndPoint section for a Serial LocalEndPoint.

PMU

The PMU section describes the characteristics of each PMU. If the RemoteEndPoint is a PDC then there may be multiple PMUs, otherwise there should be only one PMU child node per RemoteEndPoint.

- ID Code – Specifies the numeric ID of the PMU. This must match the IDCODE field sent by the C37.118 device. If there are multiple PMU nodes associated with a RemoteEndPoint (RemoteEndPoint is a PDC) with it this ID must be unique within the RemoteEndPoint collection and should match the PMUs IDCODE and not the IDCODE of the PDC itself. (`IdCode=#`, Range: 0-32,767)
- Latitude – Specifies the geographic latitude of the PMU. This field is optional. However, if the latitude is specified, then the longitude must also be specified, otherwise the interface will log an error and exit. If no sign is specified then + is assumed. (`Latitude=+/-#.#`, Range: -90.0 to +90.0)
- Longitude – Specifies the geographic longitude of the PMU. This field is optional. However, if the longitude is specified, then the latitude must also be specified, otherwise the interface will log an error and exit. If no sign is specified then + is assumed. (`Longitude=+/-#.#`, Range: -180.0 to +180.0)

PIC37118 XML Nodes, Elements, and Attributes e

Since the startup file of the C37118 interface is maintained automatically by the PI ICU, use the C37118 page to configure the startup parameters and do not make changes in the file manually. The following is the description of interface configuration parameters used in the PI ICU Control and corresponding manual parameters.

Root Node

The root node must be named PI_C37_118.Session and contains one or more LocalEndPoint leaf nodes.

PI_C37118.Session Child Nodes	Description
Example: <PI_C37_118.Session”>	
WriteIOTimeout Optional XMLType: Attribute Default: 1	By default, the interface will write the digital state I/O Timeout to all measurement type tags when a communications timeout condition is encountered. Setting the WriteIOTimeout attribute to a value of 0 will cause the interface to suppress writing I/O Timeout to the measurement tags. Valid values for this attribute are 0 (No) to not write I/O Timeout or 1 (Yes) to write I/O Timeout.
PMUErrorAction Optional XMLType: Attribute Default: 3	The PMUErrorAction attribute controls the behavior of the interface when the PMU Error flag is detected in the STAT word of the message. This flag indicates that there is a problem with the PMU hardware and that the data may not be valid. The default behavior of the interface is to discard the data. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
DataValidAction Optional XMLType: Attribute Default: 3	The DataValidAction attribute controls the behavior of the interface when the Data Invalid flag is detected in the STAT word of the message. This flag indicates that there is a problem with the data for the PMU may not be valid. This flag is typically set when the data stream is coming from a PDC versus a single PMU and is usually due to the PDC not receiving a data frame from the PMU in a reasonable time. The PDC typically blank fills the data. The default behavior of the interface is to discard the data. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values 3 – Discards the data.
SyncErrorAction Optional XMLType: Attribute Default: 1	The SyncErrorAction attribute controls the behavior of the interface when the PMU Sync Error flag is detected in the STAT word of the message. This flag indicates that there is a problem with the time synchronization of the PMU and that the quality of the timestamp is other than normal. The default behavior of the interface is to mark the data as questionable. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.

PI_C37118.Session Child Nodes	Description
Example: <PI_C37_118.Session”>	
DataSortingAction Optional XMLType: Attribute Default: 1	The DataSortingAction attribute controls the behavior of the interface when the Data Sorting flag is detected in the STAT word of the message. This error should only occur if the data stream is coming from a PDC. This flag indicates that the data for the PMU is being integrated into the data frame by arrival and not timestamp. The default behavior of the interface is to mark the data as questionable. Valid selections for this attribute are: 0 – Store the value normally (Not recommended) 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
LeapSecondAction Optional XMLType: Attribute Default: 0	The LeapSecondAction attribute controls the behavior of the interface when one of the leap second flags is detected in the STAT word of the message. This occurs when a leapsecond is occurring. The default behavior of the interface is to store the data normally. Valid selections for this attribute are: 0 – Store the value normally 1 – Mark the data as questionable. 2 – Stores the PMU Error system digital state for all measurement values. 3 – Discards the data.
SkipConfig2Cmd XML Type: Attribute Default: 0	The SkipConfig2Cmd attribute controls the behavior of interface to send CONFIG2 command to PDC / PMU each time when it is trying to reconnect to PDC / PMU. If set to yes, the interface will send CONFIG2 command to PDC / PMU only during startup. Valid values for this attribute are 0 (No) to not skip sending CONFIG2 request or 1 (Yes) to skip sending CONFIG2 request.
SendStopCmd XML Type: Attribute Default: 0	The SendStopCmd attribute controls the behavior of interface to send STOP_DATA command to the PDC / PMU when user shutdown the interface. It may affect other clients that connected to that PMU / PDC. Valid values for this attribute are 0 (No) to not send STOP_DATA command or 1 (Yes) to send STOP_DATA command when shutting down the interface.
FlushEvents XML Type: Attribute Default: 0	The FlushEvents attribute controls the behavior of interface to flush all data to the PI Data Archive after each C37.118 data messages received from PDC / PMU. If set to 0 (default value) it will flush data when buffer gets full (192 records). Valid values for this attribute are 0 (No) to not flush data or 1 (Yes) to flush all data to PI Data Archive after each data message received from PMU / PDC.
SwitchToBackupDelay XML Type: Attribute Default: 60	The SwitchToBackupDelay controls the behavior of the interface for the PDC / PMU Data Source Level Failover. If the interface is not able to set connection and receive data from primary PDC / PMU during this period of time it will switch and try to connect to the backup PDC / PMU. If it loses connection to backup and is not able to reconnect to backup PDC / PMU again during this period of time it will try to reconnect to the primary PDC / PMU again. Valid ranges: {30-600}

LocalEndPoint Nodes

Child nodes of the root node are LocalEndPoint nodes. The LocalEndPoint nodes are used to specify the properties of the local (interface side) of a physical connection such as a communication port, or TCP port. The **Type** attribute must be either `Serial` or **`IP`**. The child nodes will depend on the value of the **Type** attribute. A serial communications port will have different child nodes from an IP port. There can only be one RemoteEndPoint Node for each LocalEndPoint Node.

LocalEndPoint Child Nodes that apply to all connection types.	**Description**
Example: <RemoteEndPoint>	
Type Required XML Type: Element Default: None	Specifies the communications type. Values are “IP” or “Serial”
Id Required XMLType: Attribute Default: None	Specifies the numeric ID of the LocalEndPoint. The ID uniquely identifies the LocalEndPoint when configuring tags for an interface. The ID must be unique among all LocalEndPoints for an interface. Valid ranges: {0-65535 }
Enabled Optional XML Type: Attribute Default: 1	Enabled – Allows the LocalEndPoint and everything associated with it to be disabled and not loaded by the interface. It can then be re-enabled with having to reconfigure. A value of 0 (No) indicates that the LocalEndPoint is disabled. A value of 1 (Yes) indicates it is enabled.
CommDll Required XMLType: Element Default: None	Defines the Communications Plug-In to be used to communicate with the PDC / PMU. Due to the variation in application level functionality from one device vendor to another, the plug-in approach allows for new devices to be easily supported. Refer to Appendix C. Communication Plug-In Selection in this manual or to the OSIsoft support site for details on available plug-in’s.
DataTimeout Optional XMLType: Attribute Default: 1000 (1 second)	Controls the length of time to wait, in milliseconds, for data responses. Valid range: { 10-65535}
ReconnectRate Optional XMLType: Attribute Default: 30000 (30 seconds)	Controls the interval, in milliseconds, to be used when attempting to restart communications with a C37.118 device. Valid Range: {10-65535}
FailOverId Optional XML Type: Attribute Default: None	Specifies the numeric Failover ID of the LocalEndPoint. The Failover ID allows creating redundant pair of primary and secondary LocalEndPoints for Data Source Level Failover. Valid Range: {0-65535 or empty}
Primary Optional XML Type: Attribute Default: None	Controls which LocalEndPoint in the redundant pair becomes a primary (value should be 1). For another LocalEndPoint this field should have opposite value (0) to become a backup. Use this field to setup Data Source Level Failover. This field is disabled on ICU control if FailOverID is empty. Valid Range {0-1 or disabled}

LocalEndPoint Child Nodes that apply to all connection types.	Description
Example: <RemoteEndPoint>	
Type Required XML Type: Element Default: None	Specifies the communications type. Values are “IP” or “Serial”
LocalEndPoint Child Nodes for Type=”Serial”	**Description**
Example: <LocalEndPoint Type=”Serial”>	
Port Required XMLType: Attribute Default: None	The numeric value of the serial COM port to use. For example if using COM1 set this value to 1. Valid range : {1 – Max Ports}
Baud Optional XMLType: Attribute Default: 38400	The speed at which to communicate. Valid range: {Refer to your PDC / PMU hardware users guide for supported baud rates.}
Parity Optional XMLType: Attribute Default: None	The error checking parity to use. This can be 0 for NONE, 1 for ODD, 2 for EVEN, 3 for MARK, and 4 for SPACE. Valid range: {0,1,2,3,4}
StopBits Optional XMLType: Attribute Default: None	The number of stop bits following a message. This can be 0 for one, 1 for two, or 2 for one and a half stop bits. Valid range: {0,1,2}

LocalEndPoint Child Nodes for Type=”IP”	Description
Example: <LocalEndPoint Type=”IP”>	
DataProtocol Optional XMLType: Element Default: UDP	Specifies the protocol to use for C37.118 data messages. Valid Values: {UDP or TCP}
LocalDataPort Required if Multicast being used, otherwise optional XMLType: Attribute Default: 0 (System Assigned)	The local port to bind to for data packets. If not specified or zero the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the LocalCommandPort value or zero. Valid range: {1024-65535}
MulticastIPAddress Optional XMLType: Element Default: None	This parameter is valid only when the DataProtocol is specified as UDP. When specified, the interface will be configured to listen for C37.118 data messages via Multicast. Valid values: {224.0.0.1-224.255.255.254}

LocalEndPoint Child Nodes for Type="IP"	Description
Example: <LocalEndPoint Type="IP">	
CommandProtocol Optional XMLType: Element Default: TCP	Specifies the protocol to use for C37.118 Command and Configuration messages. If this node is not specified, the interface will use the protocol specified by the DataProtocol node or the default DataProtocol if it is not specified. Valid Values: {UDP or TCP}
LocalCommandPort Optional XMLType: Attribute Default: 0 (System Assigned)	The local port to bind to for configuration packets. If not specified or zero the interface will not specify a specific port (system assigned). This parameter will typically be used when the data protocol is UDP in order to facilitate configuration with firewalls. If this value is specified it must be different than the LocalDataPort value or zero. Valid range: {1024-65535}
CloseCommandSocket Optional XML Type: Attribute Default Value: 1	Some PMU / PDC models require the socket used for sending commands and receiving configuration and header responses be closed after use, while others require that they remain open. The Arbiter 1133 for example, requires that the socket be closed. All SEL PMU and PDC models require the socket remain open. This option only effects configurations in which the command/command response is on a separate protocol and/or port from the C37.118 data stream. Valid Values: 0 (Leave Open) or 1 (Close After Use)
LocalIPAddress Optional XMLType: Element Default: None	This parameter could be used if interface node has multiple IP addresses. When specified, the interface will bind all ports with that address. Otherwise the interface will be using default IP address. Valid Values: Any IP address that could be used by the interface node or empty.
MulticastSourceIPAddr ess Optional XML Type: Element Default Value: None	This parameter is valid only if the Local IP Address and Multicast IP Address fields have been populated. When specified, the interface will subscribe for multicast data coming from the source that has specified IP address.

RemoteEndPoint Nodes

The RemoteEndPoint node is child node of the LocalEndPoint node(s). The RemoteEndPoint node represents the remote (PDC / PMU side) of a physical connection. The RemoteEndPoint node will have one or more PMU child nodes. Each of the PMU nodes will represent a PMU that the PI C37.118 interface will communicate with through the physical port described by the LocalEndPoint / RemoteEndPoint child node pair.

RemoteEndPoint Child Nodes that apply to all connection types.	Description
Example: <RemoteEndPoint>	
IdCode Required XMLType: Attribute Default: None	Specifies the numeric ID of the PDC or PMU. This must match the IDCODE field sent by the C37.118 device. If a LocalEndPoint has multiple RemoteEndPoint nodes associated with it this ID must be unique within the LocalEndPoint collection. Valid ranges: {0-65535 }

RemoteEndPoint Child Nodes that apply to all connection types.	Description
Description Optional XML Type: Element Default: None	User defined description of the RemoteEndPoint. Ie. “Substation xyz”. This attribute is used by the interface when logging messages regarding the RemoteEndPoint.

RemoteEndPoint Child Nodes when LocalEndPoint is “Serial”	Description
Example: <RemoteEndPoint>	
	There are no Serial specific RemoteEndPoint child nodes

RemoteEndPoint Child Nodes when LocalEndPoint is “IP”	Description
Example: <RemoteEndPoint>	
HostName Required XMLType: Element Default: None	The Host name or IP address of the PDC or PMU. If using the PIC37118_Comm003.DLL, the supplied name should be set to 127.0.0.1 to indicate the interface will listen on the specified LocalDataPort for incoming C37.118 messages. Valid range: {Any valid address or name} Example: <Hostname> 192.164.55.128</Hostname> or <Hostname> myhost </Hostname>
RemoteDataPort Optional XMLType: Attribute Default: 4712	The remote port to be used for data packets. If not specified the interface will not specify a specific port (system assigned). Valid range: {1024-65535} or 0 (System Assigned)
RemoteCommandPort Optional XMLType: Attribute Default: 4713	The remote port to use for command configuration packets. Valid range: {1024-65535}

PMU Child Nodes

PMU nodes are child nodes of the RemoteEndPoint nodes. The PMU node describes the characteristics of the PMU. If the RemoteEndPoint is a PDC then there may be multiple PMUs, otherwise there should be only one PMU child node per RemoteEndPoint.

PMU Child Nodes	Description
Example: <PMU>	
IdCode Required XMLType: Attribute Default: None	Specifies the numeric ID of the PMU. This must match the IDCODE field sent by the C37.118 device. If there are multiple PMU nodes associated with a RemoteEndPoint (RemoteEndPoint is a PDC) with it this ID must be unique within the RemoteEndPoint collection and should match the PMUs IDCODE and not the PDCs. Valid ranges: {0-65535 }
Latitude Optional XMLType: Attribute Default: None	Specifies the geographic latitude of the PMU. If the latitude is specified, then the longitude must also be specified Otherwise the interface will log an error and exit. Valid range: {-90.0 to +90.0}
Longitude Optional XMLType: Attribute Default: None	Specifies the geographic longitude of the PMU. If the latitude is specified, then the longitude must also be specified Otherwise the interface will log an error and exit. Valid range: {-180.0 to +180.0}

Sample PIC37118 XML File

The following is an example file:

```
<?xml version="1.0"?>
<PI_C37_118.Session WriteIOTimeout="1" DataSortingAction="1"
PMUErrorAction="3" SyncErrorAction="1" DataValidAction="3"
LeapSecondAction="0">
   <LocalEndPoint Id="1" LocalDataPort="1024" LocalCommandPort="1025"
DataTimeout="1000">
          <Type>IP</Type>
          <CommDLL>PIC37118_Comm001.dll</CommDLL>
          <CommandProtocol>TCP</CommandProtocol>
          <DataProtocol>UDP</DataProtocol>
          <MulticastIPAddress>239.254.1.2</MulticastIPAddress>
                <RemoteEndPoint IdCode="1">
                      <Description>Arbiter in Savannah</Description>
                      <HostName>gap001</HostName>
                            <PMU IdCode="1" Latitude="37.127652"
Longitude="-122.876365"/>
                            <PMU IdCode="2"/>
                </RemoteEndPoint>
   </LocalEndPoint>
</PI_C37_118.Session>
```

Command-line Parameters

Note: The *PI Universal Interface (UniInt) User Guide* includes details about other command-line parameters, which may be useful.

Parameter	Description
`/CacheMode` Required when using disconnected startup Default: Not Defined	Required for disconnected startup operation. If defined, the **`/CacheMode`** startup parameter indicates that the interface will be configured to utilize the disconnected startup feature.
`/CachePath=`*`path`* Optional Default: Not Defined	Used to specify a directory in which to create the point caching files. The directory specified must already exist on the target machine. By default, the files are created in the same location as the interface executable. If the path contains any spaces, enclose the path in quotes. Examples: **`/CachePath=D:\PIPC\Interfaces\CacheFiles`** **`/CachePath=D:/PIPC/Interfaces/CacheFiles`** **`/CachePath=D:/PIPC/Interfaces/CacheFiles/`** Examples with space in path name: **`/CachePath="D:\Program Files\PIPC\MyFiles"`** **`/CachePath="D:/Program Files/PIPC/MyFiles"`** **`/CachePath="D:/Program Files/PIPC/MyFiles/"`**

/CacheSynch=# Optional Default: 250 ms	**NOTE:** Care must be taken when modifying this parameter. This value must be less than the smallest scan class period defined with the **/f** parameter. If the value of the **/CacheSynch** parameter is greater than the scan class value, input scans will be missed while the point cache file is being synchronized. The optional **/CacheSynch=#** startup parameter specifies the time slice period in milliseconds (ms) allocated by UniInt for synchronizing the interface point cache file with the PI Data Archive. By default, the interface will synchronize the point cache if running in the disconnected startup mode. UniInt allocates a maximum of # ms each pass through the control loop synchronizing the interface point cache until the file is completely synchronized. Synchronization of the point cache file can be disabled by setting the value **/CacheSynch=0**. The minimum synchronization period when cache synchronization is enabled is 50ms Whereas, the maximum synchronization period is 3000ms (3s). Period values of 1 to 49 will be changed by the interface to the minimum of 50ms and values greater than 3000 will be set to the maximum interval value of 3000ms. Default: 250 ms Range: {0, 50 – 3000} time in milliseconds Example: **/CacheSynch=50** (use a 50ms interval) **/CacheSynch=3000** (use a 3s interval) **/CacheSynch=0** (do not synchronize the cache)
/ec=# Optional	The first instance of the **/ec** parameter on the command-line is used to specify a counter number, #, for an I/O Rate point. If the # is not specified, then the default event counter is 1. Also, if the **/ec** parameter is not specified at all, there is still a default event counter of 1 associated with the interface. If there is an I/O Rate point that is associated with an event counter of 1, each copy of the interface that is running without **/ec=#**explicitly defined will write to the same I/O Rate point. This means either explicitly defining an event counter other than 1 for each copy of the interface or not associating any I/O Rate points with event counter 1. Configuration of I/O Rate points is discussed in the section called I/O Rate Point. For interfaces that run on Windows nodes, subsequent instances of the **/ec** parameter may be used by specific interfaces to keep track of various input operations. Subsequent instances of the **/ec** parameter can be of the form **/ec***, where * is any ASCII character sequence. For example, **/ecinput=10** and **/ec=11** are legitimate choices for the second and third event counter strings.
/f=*SS.##* or **/f=***SS.##,ss.##* or **/f=***HH:MM:SS.##* or **/f=***HH:MM:SS.##, hh:mm:ss.##* Required for reading scan-based inputs	The **/f** parameter defines the time period between scans in terms of hours (HH), minutes (MM), seconds (SS) and sub-seconds (##). The scans can be scheduled to occur at discrete moments in time with an optional time offset specified in terms of hours (hh), minutes (mm), seconds (ss) and sub-seconds (##). If HH and MM are omitted, then the time period that is specified is assumed to be in seconds. Each instance of the **/f** parameter on the command-line defines a scan class for the interface. There is no limit to the number of scan classes that can be defined. The first occurrence of the **/f** parameter on the command-line defines the first scan class of the interface; the second occurrence defines the second scan class, and so on. PI points are associated with a particular scan class via the Location4 PI point attribute. For example, all PI points that have Location4 set to 1 will receive input values at the frequency defined by the first scan class. Similarly, all points that have Location4 set

	to 2 will receive input values at the frequency specified by the second scan class, and so on.
	Two scan classes are defined in the following example:
	`/f=00:01:00,00:00:05 /f=00:00:07`
	or, equivalently:
	`/f=60,5 /f=7`
	The first scan class has a scanning frequency of 1 minute with an offset of 5 seconds, and the second scan class has a scanning frequency of 7 seconds. When an offset is specified, the scans occur at discrete moments in time according to the formula:
	scan times = (reference time) + n(frequency) + offset
	where n is an integer and the reference time is midnight on the day that the interface was started. In the above example, frequency is 60 seconds and offset is 5 seconds for the first scan class. This means that if the interface was started at 05:06:06, the first scan would be at **05:07:05**, the second scan would be at **05:08:05**, and so on. Since no offset is specified for the second scan class, the absolute scan times are undefined.
	The definition of a scan class does not guarantee that the associated points will be scanned at the given frequency. If the interface is under a large load, then some scans may occur late or be skipped entirely. See the section “Logging hit, missed, and skipped scans” in the *PI Universal Interface (UniInt) User Guide* for more information on skipped or missed scans.
	`Sub-second Scan Classes`
	Sub-second scan classes can be defined on the command-line, such as
	`/f=0.5 /f=00:00:00.1`
	where the scanning frequency associated with the first scan class is 0.5 seconds and the scanning frequency associated with the second scan class is 0.1 of a second.
	Similarly, sub-second scan classes with sub-second offsets can be defined, such as
	`/f=0.5,0.2 /f=1,0`
	`Wall Clock Scheduling`
	Scan classes that strictly adhere to wall clock scheduling are now possible. This feature is available for interfaces that run on Windows and/or UNIX. Previously, wall clock scheduling was possible, but not across daylight saving time. For example, **`/f=24:00:00,08:00:00`** corresponds to 1 scan a day starting at 8 AM. However, after a Daylight Saving Time change, the scan would occur either at 7 AM or 9 AM, depending upon the direction of the time shift. To schedule a scan once a day at 8 AM (even across daylight saving time), use **`/f=24:00:00,00:08:00,L.`** The `,L` at the end of the scan class tells UniInt to use the new wall clock scheduling algorithm.

/host=*host:port* Required	The **/host** parameter specifies the PI Data Archive node. *Host* is the IP address or the domain name of the PI Data Archive node. *Port* is the port number for TCP/IP communication. The *port* is always 5450. It is recommended to explicitly define the host and port on the command-line with the **/host** parameter. Nevertheless, if either the host or port is not specified, the interface will attempt to use defaults. Examples: The interface is running on an interface node, the domain name of the PI Data Archive node is Marvin, and the IP address of Marvin is 206.79.198.30. Valid **/host** parameters would be: **/host=marvin** **/host=marvin:5450** **/host=206.79.198.30** **/host=206.79.198.30:5450**
/id=*x* Highly Recommended	The **/id** parameter is used to specify the interface identifier. The interface identifier is a string that is no longer than 9 characters in length. UniInt concatenates this string to the header that is used to identify error messages as belonging to a particular interface. See the Appendix A. Error and Informational Messages for more information. UniInt always uses the **/id** parameter in the fashion described above. This interface also uses the **/id** parameter to identify a particular interface copy number that corresponds to an integer value that is assigned to one of the Location code point attributes, most frequently Location1. For this interface, use only numeric characters in the identifier. For example, **/id=1**
/ps=*x* Required	The **/ps** parameter specifies the point source for the interface. **X** is not case sensitive and can be any single or multiple character string. For example, **/ps=P** and **/ps=p** are equivalent. The length of **X** is limited to 100 characters by UniInt. **X** can contain any character except '*' and '?'. The point source that is assigned with the **/ps** parameter corresponds to the PointSource attribute of individual PI points. The interface will attempt to load only those PI points with the appropriate point source. If the PI API version being used is prior to 1.6.x or the PI Data Archive version is prior to 3.4.370.x, the `PointSource` is limited to a single character unless the SDK is being used.

/stopstat=*digstate* or **/stopstat** **/stopstat** only is equivalent to **/stopstat="***"ntf Shut"*" Optional Default = no digital state written at shutdown.	If **/stopstat=digstate** is present on the command line, then the digital state, `digstate`, will be written to each PI point when the interface is stopped. For a PI Data Archive, `digstate` must be in the system digital state table. . UniInt will use the first occurrence of **digstate** found in the table. If the **/stopstat** parameter is present on the startup command line, then the digital state "`Intf Shut`" will be written to each PI point when the interface is stopped. If neither **/stopstat** nor **/stopstat=digstate** is specified on the command line, then no digital states will be written when the interface is shut down. **Note:** The **/stopstat** parameter is disabled If the interface is running in a UniInt failover configuration as defined in the UniInt Failover Configuration section of this manual. Therefore, the digital state, `digstate`, will not be written to each PI point when the interface is stopped. This prevents the digital state being written to PI points while a redundant system is also writing data to the same PI points. The **/stopstat** parameter is disabled even if there is only one interface active in the failover configuration. Examples: **/stopstat=shutdown** **/stopstat="Intf Shut"** The entire `digstate` value should be enclosed within double quotes when there is a space in `digstate`.
/UFO_ID=# Required for UniInt failover phase 1 or 2	Failover ID. This value must be different from the failover ID of the other interface in the failover pair. It can be any positive, non-zero integer.
/UFO_Interval=# Optional Default: `1000` for phase 1 failover Default: `5000` for phase 2 failover Valid values are 50-20000.	Failover Update Interval Specifies the heartbeat update interval in milliseconds and must be the same on both interface computers. This is the rate at which UniInt updates the failover heartbeat points as well as how often UniInt checks on the status of the other copy of the interface.
/UFO_OtherID=# Required for UniInt failover phase 1 or 2	Other Failover ID. This value must be equal to the failover ID configured for the other interface in the failover pair.

/UFO_Sync=*path/[filename]* Required for UniInt failover phase 2 synchronization. Any valid pathname / any valid filename The default filename is generated as *executablename_pointsource_interfaceID.dat*	The failover synchronization file *path* and optional *filename* specify the path to the shared file used for failover synchronization and an optional filename used to specify a user defined filename in lieu of the default filename. The *path* to the shared file directory can be a fully qualified machine name and directory, a mapped drive letter, or a local path if the shared file is on one of the interface nodes. The *path* must be terminated by a slash (/) or backslash (\ ) character. If no d terminating slash is found, in the **/UFO_Sync** parameter, the interface interprets the final character string as an optional *filename*. The optional *filename* can be any valid filename. If the file does not exist, the first interface to start attempts to create the file. **Note:** If using the optional filename, **do not** supply a terminating slash or backslash character. If there are any spaces in the *path* or *filename*, the entire path and filename must be enclosed in quotes. **Note:** If you use the backslash and path separators and enclose the path in double quotes, the final backslash must be a double backslash (\\). Otherwise the closing double quote becomes part of the parameter instead of a parameter separator. Each node in the failover configuration must specify the same path and filename and must have read, write, and file creation rights to the shared directory specified by the *path* parameter. The service that the interface runs against must specify a valid logon user account under the “Log On” tab for the service properties.
/UFO_Type=*type* Required for UniInt failover phase 2.	The failover Type indicates which type of failover configuration the interface will run. The valid types for failover are HOT, WARM, and COLD configurations. If an interface does not supported the requested type of failover, the interface will shut down and log an error to the log file stating the requested failover type is not supported. This interface supports failover configurations COLD and HOT.

/configfile=<UNC Path> Optional Default: PIC37118.xml	This parameter specifies the interface configuration file to use to define the C37.118 device that the interface will communicate with.
/TagConfig=<UNC Path> Optional Default: None	The `/TagConfig` parameter is used to assist in creating PI tags. Because the C37.118 specification does not dictate the names to use for Phasor, Analog, and Digital channels, different PMU vendors use different names. When the interface is configured initially, the command-line parameter can be specified temporarily in the Additional Parameters test box in the ICU. When the interface is started it reads the CONFIG2 block from the PDC/PMU and generates a comma-delimited file that contains the various PI point attributes. The file can then be edited as required and then imported into PI using SMT. Because the interface exits after the file is written, it is important to remove the command-line parameter for normal operation. If a path is not specified on the command line, the generated file is saved in the interface folder. In versions of the interface earlier than 1.0.4.88, the file was saved in the system drive folder: `Windows\System32`. Optionally, you can specify the fully qualified path of where to save the taglist.csv file; for example: `/TagConfig= C:\Taglist.csv`. If the path or filename contains spaces, you must enclose the path or filename within quotation marks.

Sample PIC37118.bat File

The following is an example file:

```
REM==================================================================
REM
REM PIC37118.bat
REM
REM Sample startup file for the PI Interface for IEEE C37.118
REM
REM==================================================================
REM
REM OSIsoft strongly recommends using PI ICU to modify startup files.
REM
REM Sample command line
REM
    .\PIC37118.exe ^
    /configfile="E:\Program Files\PIPC\Interfaces\C37118\PIC37118.xml" ^
    /ps=C37118 ^
    /ID=1 ^
    /host=XXXXXX:5450
REM
REM End of pic37118.bat File
```

Chapter 9. UniInt Failover Configuration

Introduction

To minimize data loss during a single point of failure within a system, UniInt provides two failover schemes: (1) synchronization through the data source and (2) synchronization through a shared file. Synchronization through the data source is *Phase 1*, and synchronization through a shared file is *Phase 2*.

Phase 1 UniInt Failover is not supported by this interface.

Phase 2 UniInt Failover uses a shared file to synchronize failover operations and provides for *hot*, *warm*, or *cold failover*. The Phase 2 hot failover configuration provides a *no data loss* solution for a single point of failure similar to Phase 1. However, in warm and cold failover configurations, you can expect a small period of data loss during a single point of failure transition.

Note: This interface supports only Phase 2 failover, COLD and HOT configurations.

You can also configure UniInt failover to send data to a High Availability (HA) PI Data Archive collective. The collective provides redundant PI Data Archives to allow for the uninterrupted collection and presentation of PI time series data. In an HA configuration, PI Data Archives can be taken down for maintenance or repair. The HA PI Data Archive collective is described in the *High Availability Administrator Guide*.

When configured for UniInt failover, the interface routes all PI data through a state machine. The state machine determines whether to queue data or send it directly to PI depending on the current state of the interface. When the interface is in the active state, data sent through the interface gets routed directly to a PI point. In the backup state, data from the interface gets queued for a short period. Queued data in the backup interface ensures a *no-data loss* failover under normal circumstances for Phase 1 and for the hot failover configuration of Phase 2.

Quick Overview

The Quick Overview below may be used to configure this interface for failover. The failover configuration requires the two copies of the interface participating in failover be installed on different nodes. Users should verify non-failover interface operation as discussed in the Installation Checklist chapter of this manual prior to configuring the interface for failover operations. If you are not familiar with UniInt failover configuration, return to this section after reading the rest of the UniInt Failover Configuration chapter in detail. If a failure occurs at any step below, correct the error and start again at the beginning of step 6 Test in the table below. For the discussion below, the first copy of the interface configured and tested will be considered the primary interface and the second copy of the interface configured will be the backup interface.

Configuration

- One Data Source
- Two interfaces

Prerequisites

- Interface 1 is the Primary interface for collection of PI data from the data source.
- Interface 2 is the Backup interface for collection of PI data from the data source.
- You must set up a shared file if using Phase 2 failover.
- Phase 2: The shared file must store data for five failover tags:

  (1) Active ID.

  (2) Heartbeat 1.

  (3) Heartbeat 2.

  (4) Device Status 1.

  (5) Device Status 2.
- Each interface must be configured with two required failover command line parameters: (1) its FailoverID number (**`/UFO_ID`**); (2) the FailoverID number of its Backup interface (**`/UFO_OtherID`**). You must also specify the name of the PI Data Archive host for exceptions and PI point updates.
- All other configuration parameters for the two interfaces must be identical.

Synchronization through a Shared File (Phase 2)

Figure 1: Synchronization through a Shared File (Phase 2) Failover Architecture

The Phase 2 failover architecture is shown in Figure 2 which depicts a typical network setup including the path to the synchronization file located on a File Server (FileSvr). Other configurations may be supported and this figure is used only as an example for the following discussion.

For a more detailed explanation of this synchronization method, see Detailed Explanation of Synchronization through a Shared File (Phase 2)

Configuring Synchronization through a Shared File (Phase 2)

<table>
<tr><th>Step</th><th>Description</th></tr>
<tr><td>1.</td><td>Verify non-failover interface operation as described in the Installation Checklist section of this manual</td></tr>
<tr><td>2.</td><td>Configure the Shared File
Choose a location for the shared file. The file can reside on one of the interface nodes or on a separate node from the interfaces; however OSIsoft strongly recommends that you put the file on a Windows Server platform that has the “File Server” role configured. .
Setup a file share and make sure to assign the permissions so that both primary and backup interfaces have read/write access to the file.</td></tr>
<tr><td>3.</td><td>Configure the interface parameters
Use the Failover section of the interface Configuration Utility (ICU) to enable failover and create two parameters for each interface: (1) a Failover ID number for the interface; and (2) the Failover ID number for its backup interface.
The Failover ID for each interface must be unique and each interface must know the Failover ID of its backup interface.
If the interface can perform using either Phase 1 or Phase 2 pick the Phase 2 radio button in the ICU.
Select the synchronization File Path and File to use for Failover.
Select the type of failover required (Cold, Warm, Hot). The choice depends on what types of failover the interface supports.
Ensure that the user name assigned in the “Log on as:” parameter in the Service section of the ICU is a user that has read/write access to the folder where the shared file will reside.
All other command line parameters for the primary and secondary interfaces must be identical.
If you use a PI Data Archive Collective, you must point the primary and secondary interfaces to different members of the collective by setting the SDK Member under the PI Host Information section of the ICU.
[Option] Set the update rate for the heartbeat point if you need a value other than the default of 5000 milliseconds.</td></tr>
<tr><td>4.</td><td>Configure the PI points
Configure five PI points for the interface: the active ID, heartbeat 1, heartbeat 2, device status 1 and device status 2. You can also configure two state points for monitoring the status of the interfaces.
Do not confuse the failover device status points with the UniInt health device status points. The information in the two points is similar, but the failover device status points are integer values and the health device status points are string values.
<table>
<tr><th>Tag</th><th>ExDesc</th><th>digitalset</th><th></th></tr>
<tr><td>ActiveID</td><td>[UFO2_ACTIVEID]</td><td></td><td rowspan="7">UniInt does not examine the remaining attributes, but the PointSource and Location1 must match.</td></tr>
<tr><td>IF1_Heartbeat
(IF-Node1)</td><td>[UFO2_HEARTBEAT:#]</td><td></td></tr>
<tr><td>IF2_Heartbeat
(IF-Node2)</td><td>[UFO2_HEARTBEAT:#]</td><td></td></tr>
<tr><td>IF1_DeviceStatus
(IF-Node1)</td><td>[UFO2_DEVICESTAT:#]</td><td></td></tr>
<tr><td>IF2_DeviceStatus
(IF-Node2)</td><td>[UFO2_DEVICESTAT:#]</td><td></td></tr>
<tr><td>IF1_State
(IF-Node1)</td><td>[UFO2_STATE:#]</td><td>IF_State</td></tr>
<tr><td>IF2_State
(IF-Node2)</td><td>[UFO2_STATE:#]</td><td>IF_State</td></tr>
</table></td></tr>
<tr><td>5.</td><td>Test the configuration.</td></tr>
</table>

Step	Description
	After configuring the shared file and the interface and PI points, the interface should be ready to run. For help resolving failover file issues, see OSIsoft KB article - KB00889 - Troubleshooting Failover Synchronization File Errors on the OSIsoft technical support web site. See Troubleshooting UniInt Failover for help resolving Failover issues. 1. Start the primary interface interactively without buffering. 2. Verify a successful interface start by reviewing the log file. The log file will contain messages that indicate the failover state of the interface. A successful start with only a single interface copy running will be indicated by an informational message stating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface not available.`" If the interface has failed to start, an error message will appear in the log file. For details relating to informational and error messages, refer to Appendix A. Error and Informational Messages. 3. Verify data on the PI Data Archive using available PI tools. • The active ID control point in the PI Data Archive must be set to the value of the running copy of the interface as defined by the **`/UFO_ID`** startup command-line parameter. • The heartbeat control point in the PI Data Archive must be changing values at a rate specified by the **`/UFO_Interval`** startup command-line parameter. 4. Stop the primary interface. 5. Start the backup interface interactively without buffering. Notice that this copy will become the primary because the other copy is stopped. 6. Repeat steps 2, 3, and 4. 7. Stop the backup interface. 8. Start buffering. 9. Start the primary interface interactively. 10. Once the primary interface has successfully started and is collecting data, start the backup interface interactively. 11. Verify that both copies of the interface are running in a failover configuration. • Review the log file for the copy of the interface that was started first. The log file will contain messages that indicate the failover state of the interface. The state of this interface must have changed as indicated with an informational message stating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface available.`" If the interface has not changed to this state, browse the log file for error messages. For details relating to informational and error messages, refer to Appendix A. Error and Informational Messages. • Review the log file for the copy of the interface that was started last. The log file will contain messages that indicate the failover state of the interface. A successful start of the interface will be indicated by an informational message stating "`UniInt failover: Interface in the "Backup" state.`" If the interface has failed to start, an error message will appear in the log file. For details relating to informational and error messages, refer to the Appendix A. Error and Informational Messages section below. 12. Verify data in the PI Data Archive using available PI tools.

Step	Description
	• The active ID control point in the PI Data Archive must be set to the value of the running copy of the interface that was started first as defined by the **`/UFO_ID`** startup command-line parameter. • The heartbeat control points for both copies of the interface in the PI Data Archive must be changing values at a rate specified by the **`/UFO_Interval`** startup command-line parameter or the scan class which the points have been built against. 13. Test Failover by stopping the primary interface. 14. Verify the backup interface has assumed the role of primary by searching the `pipc.log` file for a message indicating the backup interface has changed to the "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface not available`." The backup interface is now considered primary and the previous primary interface is now backup. 15. Verify data in the PI Data Archive. For hot failover, there may be an overlap of data due to the queuing of data, but there must be no data loss. For warm or cold failover, short gaps in archived data are expected. 16. Start the backup interface. Once the primary interface detects a backup interface, the primary interface will now change state indicating "`UniInt failover: Interface in the "Primary" state and actively sending data to PI. Backup interface available`." In the `pipc.log` file. 17. Verify the backup interface starts and assumes the role of backup. A successful start of the backup interface will be indicated by an informational message stating "`UniInt failover: Interface in "Backup state.`" Since this is the initial state of the interface, the informational message will be near the beginning of the start sequence of the `pipc.log` file. 18. Test failover with different failure scenarios (for example, loss of the PI Data Archive connection for a single interface copy). UniInt hot failover guarantees no data loss with a single point of failure; verify no data loss by checking the data in the PI Data Archive and on the data source. For warm or cold failover, short gaps in archived data can occur. 19. Stop both copies of the interface, start buffering, start each interface as a service. 20. Verify data as stated above. 21. To designate a specific interface as primary, set the active ID point to the ID of the desired primary interface as defined by the **`/UFO_ID`** startup command-line parameter.

Configuring UniInt Failover through a Shared File (Phase 2)

Start-Up Parameters

Note: The **`/stopstat`** parameter is disabled if the interface is running in a UniInt failover configuration. Therefore, the digital state, `digstate`, will not be written to each PI point when the interface is stopped. This prevents the digital state being written to PI points while a redundant system is also writing data to the same PI points. The **`/stopstat`** parameter is disabled even if there is only one interface active in the failover configuration.

The following table lists the start-up parameters used by UniInt Failover Phase 2. All of the parameters are required except the **`/UFO_Interval`** startup parameter. See the table below for further explanation.

Parameter	Required/ Optional	Description	Value/Default
`/UFO_ID=`*`#`*	Required	Failover ID for IF-Node1 This value must be different from the failover ID of IF-Node2.	Any positive, non-zero integer / **1**
	Required	Failover ID for IF-Node2 This value must be different from the failover ID of IF-Node1.	Any positive, non-zero integer / **2**
`/UFO_DNFBPI`	Optional	If this parameter is enabled and both the primary and backup instances lose connection to the PI Data Archive UniInt will not failover and incoming data will be buffered by a single interface instance to prevent the buffer from being split.	Undefined
`/UFO_OtherID=`*`#`*	Required	Other Failover ID for IF-Node1 The value must be equal to the Failover ID configured for the interface on IF-Node2.	Same value as Failover ID for IF-Node2 / **2**
	Required	Other Failover ID for IF-Node2 The value must be equal to the Failover ID configured for the interface on IF-Node1.	Same value as Failover ID for IF-Node1 / **1**
`/UFO_Sync=` *`path/[filename]`*	Required for Phase 2 synchronization	The Failover File Synchronization file *path* and optional *filename* specify the path to the shared file used for failover synchronization and an optional filename used to specify a user defined filename in lieu of the default filename. The *path* to the shared file directory can be a fully qualified machine name and directory, a mapped drive letter, or a local path if the shared file is on one of the interface nodes. The *path* must be terminated by a slash (/) or backslash (\ ) character.	Any valid pathname / any valid filename The default filename is generated as *executablename_ pointsource_ interfaceID.dat*

Parameter	Required/ Optional	Description	Value/Default
		If no terminating slash is found, in the **`/UFO_Sync`** parameter, the interface interprets the final character string as an optional *filename*. The optional *filename* can be any valid filename. If the file does not exist, the first interface to start attempts to create the file. **Note:** If using the optional filename, **do not** supply a terminating slash or backslash character. If there are any spaces in the *path* or *filename*, the entire path and filename must be enclosed in quotes. **Note:** If you use the backslash and path separators and enclose the path in double quotes, the final backslash must be a double backslash (`\\`). Otherwise the closing double quote becomes part of the parameter instead of a parameter separator. Each node in the failover configuration must specify the same path and filename and must have read, write, and file creation rights to the shared directory specified by the *path* parameter. The service that the interface runs against must specify a valid logon user account under the “Log On” tab for the service properties.	
`/UFO_Type=`*`type`*	Required	The Failover Type indicates which type of failover configuration the interface will run. The valid types for failover are HOT, WARM, and COLD configurations. If an interface does not supported the requested type of failover, the interface will shutdown and log an error to the `pipc.log` file stating the requested failover type is not supported.	COLD|WARM|HOT / **COLD**

Parameter	Required/ Optional	Description	Value/Default
/UFO_Interval=*#*	Optional	Failover Update Interval Specifies the heartbeat Update Interval in milliseconds and must be the same on both interface computers. This is the rate at which UniInt updates the failover heartbeat points as well as how often UniInt checks on the status of the other copy of the interface.	50 – 20000 / **5000**
/Host=*server*	Required	Host PI Data Archive for exceptions and PI point updates The value of the **/Host** startup parameter depends on the PI Data Archive configuration. If the PI Data Archive is not part of a collective, the value of **/Host** must be identical on both interface computers. If the redundant interfaces are being configured to send data to a PI Data Archive collective, set the value of the **/Host** parameters on the different interface nodes to different members of the collective.	For IF-Node1 PrimaryPI / **None** For IF-Node2 SecondaryPI / **None**

Failover Control Points

The following table describes the points that are required to manage failover. In Phase 2 Failover, these points are located in a data file shared by the primary and backup interfaces.

OSIsoft recommends that you locate the shared file on a dedicated server that has no other role in data collection. This avoids potential resource contention and processing degradation if your system monitors a large number of data points at a high frequency.

Point	Description	Value / Default
ActiveID	Monitored by the interfaces to determine which interface is currently sending data to the PI Data Archive. **ActiveID** must be initialized so that when the interfaces read it for the first time, it is not an error. **ActiveID** can also be used to force failover. For example, if the current primary is IF-Node 1 and **ActiveID** is 1, you can manually change **ActiveID** to 2. This causes the interface at IF-Node2 to transition to the primary role and the interface at IF-Node1 to transition to the backup role.	From 0 to the highest interface Failover ID number / **None**) Updated by the redundant interfaces Can be changed manually to initiate a manual failover
Heartbeat 1	Updated periodically by the interface on IF-Node1. The interface on IF-Node2 monitors this value to determine if the interface on IF-Node1 has become unresponsive.	Values range between 0 and 31 / **None** Updated by the interface on IF-Node1

Point	Description	Value / Default
Heartbeat 2	Updated periodically by the interface on IF-Node2. The interface on IF-Node1 monitors this value to determine if the interface on IF-Node2 has become unresponsive.	Values range between 0 and 31 / **None** Updated by the interface on IF-Node2

PI Points

The following tables list the required UniInt failover control PI points, the values they will receive, and descriptions.

Active_ID Point Configuration

Attributes	ActiveID
Tag	**<Intf>_ActiveID**
CompMax	**0**
ExDesc	[UFO2_ActiveID]
Location1	Match # in **/id=#**
Location5	Optional, Time in min to wait for backup to collect data before failing over.
PointSource	Match *x* in **/ps=***x*
PointType	Int32
Shutdown	0
Step	1

Heartbeat and Device Status Point Configuration

Attribute	Heartbeat 1	Heartbeat 2	DeviceStatus 1	DeviceStatus 2
Tag	<HB1>	<HB2>	<DS1>	<DS2>
ExDesc	[UFO2_Heartbeat: #] Match # in **/UFO_ID=#**	[UFO2_Heartbeat: #] Match # in **/UFO_OtherID=#**	[UFO2_DeviceStat: #] Match # in **/UFO_ID=#**	[UFO2_DeviceStat: #] Match # in **/UFO_OtherID=#**
Location1	Match # in **/id=#**	Match # in **/id=#**	Match # in **/id=#**	Match # in **/id=#**
Location5	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.	Optional, Time in min to wait for backup to collect data before failing over.
Point Source	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*	Match *x* in **/ps=***x*
PointType	int32	int32	int32	int32
Shutdown	0	0	0	0
Step	1	1	1	1

Interface State Point Configuration

Attribute	Primary	Backup
Tag	<Tagname1>	<Tagname2>
CompMax	0	0

Attribute	Primary	Backup
DigitalSet	UFO_State	UFO_State
ExDesc	[UFO2_State:#] (Match **/UFO_ID=**# on primary node)	[UFO2_State:#] (Match **/UFO_ID=**# on backup node)
Location1	Match # in **/id=**#	Same as for primary node
PointSource	Match *x* in **/ps=***x*	Same as for primary node
PointType	digital	digital
Shutdown	0	0
Step	1	1

The following table describes the extended descriptor for the above PI points in more detail.

PI Tag ExDesc	Required / Optional	Description	Value
[UFO2_ACTIVEID]	Required	Active ID point The ExDesc must start with the case sensitive string: [UFO2_ACTIVEID]. The PointSource must match the interfaces' Pointsource. Location1 must match the ID for the interfaces. Location5 is the COLD failover retry interval in minutes. This can be used to specify how long before an interface retries to connect to the device in a COLD failover configuration. (See the description of COLD failover retry interval for a detailed explanation.)	0 – highest Interface Failover ID Updated by the redundant interfaces
[UFO2_HEARTBEAT:#] (IF-Node1)	Required	Heartbeat 1 point The ExDesc must start with the case sensitive string: [UFO2_HEARTBEAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node1. The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces.	0 – 31 / **None** Updated by the interface on IF-Node1
[UFO2_HEARTBEAT:#] (IF-Node2)	Required	Heartbeat 2 point The ExDesc must start with the case sensitive string: [UFO2_HEARTBEAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2. The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces.	0 – 31 / **None** Updated by the interface on IF-Node2

PI Tag ExDesc	Required / Optional	Description	Value
[UFO2_DEVICESTAT :#] (IF-Node1)	Required	Device Status 1 Tag The ExDesc must start with the case sensitive string: [UFO2_DEVICESTAT:#] The value following the colon (☺ must be the failover ID for the interface running on IF-Node1 The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces. A lower value is a better status and the interface with the lower status will attempt to become the primary interface. The failover 1 device status point is very similar to the UniInt Health Device Status point except the data written to this point are integer values. A value of 0 is good and a value of 99 is OFF. Any value between these two extremes may result in a failover. The interface client code updates these values when the health device status point is updated.	0 – 99 / **None** Updated by the interface on IF-Node1
[UFO2_DEVICESTAT :#] (IF-Node2)	Required	Device Status 2 point The ExDesc must start with the case sensitive string: [UFO2_DEVICESTAT:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2 The PointSource must match the interfaces' /ps parameter. Location1 must match the ID for the interfaces. A lower value is a better status and the interface with the lower status will attempt to become the primary interface.	0 – 99 / **None** Updated by the interface on IF-Node2
[UFO2_STATE:#] (IF-Node1)	Optional	State 1 point The ExDesc must start with the case sensitive string: [UFO2_STATE:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node1 The failover state point is recommended. The failover state points are digital points assigned to a digital state set with the following values. 0 = Off: The interface has been shut down.	0 – 5 / **None** Normally updated by the interface currently in the primary role.

PI Tag ExDesc	Required / Optional	Description	Value
		1 = Backup No Data Source: The interface is running but cannot communicate with the data source. 2 = Backup No PI Connection: The interface is running and connected to the data source but has lost its communication to the PI Data Archive. 3 = Backup: The interface is running and collecting data normally and is ready to take over as primary if the primary interface shuts down or experiences problems. 4 = Transition: The interface stays in this state for only a short period of time. The transition period prevents thrashing when more than one interface attempts to assume the role of primary interface. 5 = Primary: The interface is running, collecting data and sending the data to the PI Data Archive.	
[UFO2_STATE:#] (IF-Node2)	Optional	State 2 point The ExDesc must start with the case sensitive string: [UFO2_STATE:#] The number following the colon (☺ must be the Failover ID for the interface running on IF-Node2 The failover state point is recommended.	Normally updated by the interface currently in the Primary state. Values range between 0 and 5. See description of State 1 point.

Detailed Explanation of Synchronization through a Shared File (Phase 2)

In a shared file failover configuration, there is no direct failover control information passed between the data source and the interface. This failover scheme uses five PI tags to control failover operation, and all failover communication between primary and backup interfaces passes through a shared data file.

Once the interface is configured and running, the ability to read or write to the PI tags is not required for the proper operation of failover (unless connection to the shared file is lost). This solution does not require a connection to the PI Data Archive after initial startup because the control point data are set and monitored in the shared file. However, the PI point values are sent to the PI Data Archive so that you can monitor them with standard OSIsoft client tools.

You can force manual failover by changing the **ActiveID** on the PI Data Archive to the backup failover ID.

The preceding figure shows a typical network setup in the normal or steady state. The solid magenta lines show the data path from the interface nodes to the shared file used for failover synchronization. The shared file can be located anywhere in the network as long as both interface nodes can read, write, and create the necessary file on the shared file machine. OSIsoft strongly recommends that you put the file on a dedicated file server that has no other role in the collection of data.

The major difference between synchronizing the interfaces through the data source (Phase 1) and synchronizing the interfaces through the shared file (Phase 2) is where the control data is located. When synchronizing through the data source, the control data is acquired directly from the data source. We assume that if the primary interface cannot read the failover control points, then it cannot read any other data. There is no need for a backup communications path between the control data and the interface.

When synchronizing through a shared file, however, phase 2 cannot assume that loss of control information from the shared file implies that the primary interface is down. We must account for the possible loss of the path to the shared file itself and provide an alternate control path to determine the status of the primary interface. For this reason, if the shared file is unreachable for any reason, the interfaces use the PI Data Archive as an alternate path to pass control data.

When the backup interface does not receive updates from the shared file, it cannot tell definitively why the primary is not updating the file, whether the path to the shared file is down, whether the path to the data source is down, or whether the interface itself is having problems. To resolve this uncertainty, the backup interface uses the path to the PI Data Archive to determine the status of the primary interface. If the primary interface is still communicating with the PI Data Archive, then failover to the backup is not required. However, if the primary interface is not posting data to the PI Data Archive, then the backup must initiate failover operations.

The primary interface also monitors the connection with the shared file to maintain the integrity of the failover configuration. If the primary interface can read and write to the shared file with no errors but the backup control information is not changing, then the backup is experiencing some error condition. To determine exactly where the problem exists, the primary interface uses the path to the PI Data Archive to establish the status of the backup interface. For example, if the backup interface information indicates that it has been shutdown, it may have been restarted and is now experiencing errors reading and writing to the shared file. Both primary and backup interfaces must always check their status through the PI Data Archive to determine if one or the other is not updating the shared file and why.

Steady State Operation

Steady state operation is considered the normal operating condition. In this state, the primary interface is actively collecting data and sending its data to PI points. The primary interface is also updating its heartbeat value, monitoring the heartbeat value for the backup interface, checking the active ID value, and checking the device status for the backup interface every failover update interval on the shared file. Likewise, the backup interface is updating its heartbeat value, monitoring the heartbeat value for the primary interface, checking the active ID value, and checking the device status for the primary interface every failover update interval on the shared file. As long as the heartbeat value for the primary interface indicates that it is operating properly, the active ID has not changed, and the device status on the primary interface is good, the backup interface will continue in this mode of operation.

An interface configured for hot failover will have the backup interface actively collecting and queuing data but not sending that data to the PI Data Archive. An interface for warm failover in the backup role is not actively collecting data from the data source even though it may be configured with PI points and may even have a good connection to the data source. An interface configured for cold failover in the backup role is not connected to the data source and upon initial startup will not have configured PI points.

The interaction between the interface and the shared file is fundamental to failover. The discussion that follows only refers to the data written to the shared file. However, every value written to the shared file is echoed to the points in the PI Data Archive. Updating of the points in the PI Data Archive is assumed to take place unless communication with the PI Data Archive is interrupted. The updates to the PI Data Archive will be buffered by Bufserv or PIBufss in this case.

In a hot failover configuration, each interface participating in the failover solution will queue three failover intervals worth of data to prevent any data loss. When a failover occurs, there may be a period of overlapping data for up to three intervals. The exact amount of overlap is determined by the timing and the cause of the failover and may be different every time. Using the default update interval of 5 seconds will result in overlapping data between 0 and 15 seconds. The no data loss claim for hot failover is based on a single point of failure. If both interfaces have trouble collecting data for the same period of time, data will be lost during that time.

As mentioned above, each interface has its own heartbeat value. In normal operation, the heartbeat value in the shared file is incremented by UniInt from 1 to 15 and then wraps around to a value of 1 again. UniInt increments the heartbeat value in the shared file every failover update interval. The default failover update interval is 5 seconds. UniInt also reads the heartbeat value for the other interface copy participating in failover every failover update interval. If the connection to the PI Data Archive is lost, the value of the heartbeat will be incremented from 17 to 31 and then wrap around to a value of 17 again. Once the connection to the PI Data Archive is restored, the heartbeat values will revert back to the 1 to 15 range. During a normal shutdown process, the heartbeat value will be set to zero.

During steady state, the active ID will equal the value of the failover ID of the primary interface. This value is set by UniInt when the interface enters the primary state and is not updated again by the primary interface until it shuts down gracefully. During shutdown, the primary interface will set the active ID to zero before shutting down. The backup interface has the ability to assume control as primary even if the current primary is not experiencing problems. This can be accomplished by setting the active ID point in the PI Data Archive to the active ID of the desired interface copy.

As previously mentioned, in a hot failover configuration the backup interface actively collects data but does not send its data to PI points. To eliminate any data loss during a failover, the backup interface queues data in memory for three failover update intervals. The data in the queue is continuously updated to contain the most recent data. Data older than three update intervals is discarded if the primary interface is in a good status as determined by the backup. If the backup interface transitions to the primary, it will have data in its queue to send to the PI points. This queued data is sent to the PI points using the same function calls that would have been used had the interface been in a primary state when the data was collected. If UniInt receives data without a timestamp, the primary copy uses the current PI time to timestamp data sent to PI points. Likewise, the backup copy timestamps data it receives without a timestamp with the current PI time before queuing its data. This preserves the accuracy of the timestamps.

Failover Configuration Using PI ICU

The use of the PI ICU is the recommended and safest method for configuring the interface for UniInt failover. With the exception of the notes described in this section, the interface shall be configured with the PI ICU as described in the Configuring the Interface with PI ICU section of this manual.

Note: With the exception of the **`/UFO_ID`** and **`/UFO_OtherID`** startup command-line parameters, the UniInt failover scheme requires that both copies of the interface have identical startup command files. This requirement causes the PI ICU to produce a message when creating the second copy of the interface stating that the "PS/ID combo is already in use by the interface" as shown in

Figure below. Ignore this message and click the *Add* button.

Create the Interface Instance with PI ICU

If the interface does not already exist in the ICU it must first be created. The procedure for doing this is the same as for non-failover interfaces. When configuring the second instance for UniInt Failover the *Point Source* and *Interface ID #* boxes will be in yellow and a message will be displayed saying this is already in use. This should be ignored.

Figure 4: PI ICU configuration screen shows that the "PS/ID combo is already in use by the interface." The user must ignore the yellow boxes, which indicate errors, and click the *Add* button to configure the interface for failover.

Configuring the UniInt Failover Startup Parameters with PI ICU

There are three interface startup parameters that control UniInt failover: **`/UFO_ID`**, **`/UFO_OtherID`**, and **`/UFO_Interval`**. The `UFO` stands for UniInt Failover. The **`/UFO_ID`** and **`/UFO_OtherID`** parameters are required for the interface to operate in a failover configuration, but the **`/UFO_Interval`** is optional. For more detail on each of these parameters, see the Configuring UniInt Failover through a Shared File (Phase 2) Start-Up Parameters section of this manual.

Figure 5: This figure illustrates the PI ICU failover configuration screen showing the UniInt failover startup parameters (Phase 2). This copy of the interface defines its Failover ID as 2 (**`/UFO_ID=2`**) and the other interfaces Failover ID as 1 (**`/UFO_OtherID=1`**). The other failover interface copy must define its Failover ID as 1 (**`/UFO_ID=1`**) and the other interface Failover ID as 2 (**`/UFO_OtherID=2`**) in its ICU failover configuration screen. It also defines the location and name of the synchronization file as well as the type of failover as COLD.

Creating the Failover State Digital State Set

The UFO_State digital state set is used in conjunction with the failover state digital point. If the UFO_State digital state set has not been created yet, it can be created using either the *Failover* page of the ICU (1.4.1.0 or later) or the Digital States plug-in in the SMT 3 Utility (3.0.0.7 or later).

Using the PI ICU Utility to create Digital State Set

To use the UniInt *Failover* page to create the UFO_State digital state set, right-click on any of the failover points listed in the *Tag* column and then click the *Create UFO_State Digital Set on Server XXXXXX…* command, where *XXXXXX* is the PI Data Archive where the points will be or are created.

This command is unavailable if the UFO_State digital state set already exists on the *XXXXXX* PI Data Archive.

Using the PI SMT 3 Utility to create Digital State Set

Optionally the *Export UFO_State Digital Set (.csv)* command on the shortcut menu can be selected to create a comma-separated file to be imported via the System Management Tools (SMT3) (version 3.0.0.7 or higher) or use the `UniInt_Failover_DigitalSet_UFO_State.csv` file included in the installation kit.

The procedure below outlines the steps necessary to create a digital set on a PI Data Archive using the *Import from File* command found in the SMT3 application. The procedure assumes the user has a basic understanding of the SMT3 application.

1. Open the SMT3 application.
2. Select the appropriate PI Data Archive from the PI Data Archives window. If the desired server is not listed, add it using the PI Connection Manager. A view of the SMT application is shown in Figure 7 below.
3. From the *System Management Plug-Ins* window, expand *Points* then select *Digital States*. A list of available digital state sets will be displayed in the main window for the selected PI Data Archive. Refer to Figure 7 below.
4. In the main window, right-click on the desired server and select the *Import from File* command. Refer to Figure 7 below.

Figure 7: PI SMT application configured to import a digital state set file. The PI Data Archives window shows the "localhost" PI Data Archive selected along with the System Management Plug-Ins window showing the Digital States Plug-In as being selected. The digital state set file can now be imported by selecting the *Import from File* command.

5. Navigate to and select the `UniInt_Failover_DigitalSet_UFO_State.csv` file for import using the Browse icon on the display. Select the desired *Overwrite Options*. Refer to Figure below.

Figure 8: PI SMT application *Import Digital Set(s)* window. This view shows the `UniInt_Failover_DigitalSet_UFO_State.csv` file as being selected for import. Select the desired *Overwrite Options* by choosing the appropriate option button.

6. Click on the *OK* button. Refer to Figure above.

7. The `UFO_State` digital set is created as shown in Figure below.

Figure 9: The PI SMT application showing the `UFO_State` digital set created on the "localhost" PI Data Archive.

Creating the UniInt Failover Control and Failover State Points (Phase 2)

The ICU can be used to create the UniInt failover control and state points.

To use the ICU *Failover* page to create these points, simply right-click any of the failover points in the list and click the *Create all points (UFO Phase 2)* command.

If this menu command is unavailable, it is because the UFO_State digital state set has not been created in the PI Data Archive yet. *Create UFO_State Digital Set on Server xxxxxxx…* on the shortcut menu can be used to create that digital state set. After this has been done then the *Create all points (UFO Phase2)* command should be available.

Once the failover control and failover state tags have been created the *Failover* page of the ICU should look similar to the illustration below.

Chapter 10. Interface Node Clock

Make sure that the time and time zone settings on the computer are correct. To confirm, run the Date/Time applet located in the Windows Control Panel. If the locale where the interface node resides observes Daylight Saving Time, check the *Automatically adjust clock for daylight saving changes* box. For example,

In addition, make sure that the TZ environment variable is not defined. All of the currently defined environment variables can be viewed by opening a Command Prompt window and typing `set`. That is,

```
C:> set
```

Confirm that TZ is not in the resulting list. If it is, run the System applet of the Control Panel, click the *Environment Variables* button under the *Advanced* tab, and remove TZ from the list of environment variables.

Chapter 11. Security

Windows

The PI Firewall Database and the PI Trust Database must be configured so that the interface is allowed to write data to the PI Data Archive.

The Trust Database, which is maintained by the PI Base Subsystem, replaces the Proxy Database used prior to PI Data Archive version 3.3. The PI Trust Database maintains all the functionality of the proxy mechanism while being more secure.

See "Manage Interface Authentication with PI Trusts" in the chapter "Manage Security" of the *PI Server Introduction to System Management Guide*.

If the interface cannot write data to the PI Data Archive because it has insufficient privileges, a `-10401` error will be reported in the log file. If the interface cannot send data to a PI2 Data Archive, it writes a -999 error. See the section Appendix A. Error and Informational Messages for additional information on error messaging.

Authentication

Interface instances are usually configured to run as Windows services. Since a service runs in a non-interactive context, a PI trust is required to authenticate the interface service to the PI Data Archive. A PI trust is associated with one PI identity, PI user, or PI group. When an interface successfully authenticates through a trust, the interface is granted the access rights for the associated identity, user, or group.

OSIsoft discourages using highly-privileged identities, users, or groups in PI trusts for interfaces.

Security Note: Avoid using the piadmin super-user and piadmins group. The recommended best practice for PI Data Archive security is to create an identity, user, or group that has only the access rights which are necessary for the interface to operate.

PI Data Archive v3.3 and Later

Security Configuration using Trust Editor

The Trust Editor plug-in for PI System Management Tools edits the PI trust table.

See the "Manage Interface Authentication with PI Trusts" section in the PI Server Introduction to System Management manual for more details on security configuration.

Security configuration using piconfig

For PI Data Archive v3.3 and higher, the following example demonstrates how to edit the PI trust table with ipconfig.

```
C:\PI\adm> ipconfig
@table pitrust
@mode create
@istr Trust,IPAddr,NetMask,PIUser
a_trust_name,192.168.100.11,255.255.255.255,trust_identity
@quit
```

For the preceding example,

`Trust`: An arbitrary name for the trust table entry; in the above example,

`a_trust_name`

`IPAddr`: the IP address of the computer running the interface; in the above example,

`192.168.100.11`

`NetMask`: the network mask; *`255.255.255.255`* specifies an exact match with `IPAddr`

`PIUser`: the PI identity, user, or group the interface is entrusted as; in the example,

`trust_identity`

PI Data Archive v3.2

For PI Data Archive v3.2, the following example demonstrates how to edit the PI Proxy table:

```
C:\PI\adm> p100ointed100@table pi_gen,piproxy
@mode create
@istr host,proxyaccount
piapimachine,piadmin
@quit
```

In place of piapimachine, put the name of the interface node as it is seen by the PI Data Archive.

Authorization

For an interface instance to start and write data to PI points, the following permissions must be granted to the PI identity, user, or group in the PI trust that authenticates the interface instance.

Database Security	Permission	Notes
PIPOINT	r	

Point Database	Permission	Notes
PtSecurity	r	
DataSecurity	r,w	Unbuffered
	r	Buffered (the buffering application requires r,w for the interface points)

The permissions in the preceding table must be granted for every PI point that is configured for the interface instance. Observe that buffering on the interface node is significant to PI point permissions.

When the interface instance is running on an unbuffered interface node, the interface instance sends PI point updates directly to the PI Data Archive. Therefore, the DataSecurity attribute must grant write access to the PI identity, user, or group in the PI trust that authenticates the interface instance.

When the interface instance is running on a buffered interface node, the interface instance sends PI point updates to the local buffering application, which relays the PI point updates to the PI Data Archive. The buffering application is a separate client to the PI Data Archive and, therefore, authenticates independently of the interface instances. The DataSecurity attribute must grant write access to the PI identity, user, or group in the PI trust that authenticates the buffering application.

Chapter 12. Starting / Stopping the Interface

This section describes starting and stopping the interface once it has been installed as a service. See the *PI Universal Interface (UniInt) User Guide* to run the interface interactively.

Starting Interface as a Service

If the interface was installed as service, it can be started from PI ICU, the Services control panel or with the command:

```
PIC37118.exe / start [ /serviceid id ]
```

To start the interface service with PI ICU, use the button on the PI ICU toolbar.

A message will inform the user of the status of the interface service. Even if the message indicates that the service has started successfully, double check through the Services control panel applet. Services may terminate immediately after startup for a variety of reasons, and one typical reason is that the service is not able to find the command-line parameters in the associated .bat file. Verify that the root name of the .bat file and the .exe file are the same, and that the .bat file and the .exe file are in the same directory. Further troubleshooting of services might require consulting the log file, Windows Event Viewer, or other sources of log messages. See the section Appendix A. Error and Informational Messages for additional information.

Stopping Interface Running as a Service

If the interface was installed as service, it can be stopped at any time from PI ICU, the Services control panel or with the command:

```
PIC37118.exe /stop [ /serviceid id ]
```

The service can be removed by:

```
PIC37118.exe /remove [ /serviceid id ]
```

To stop the interface service with PI ICU, use the button on the PI ICU toolbar.

Chapter 13. Buffering for PI Interfaces

The interface node uses buffering to prevent data loss when PI Data Archive is not available. UniInt interfaces can buffer data to store point values when network communication to the PI Data Archive is unavailable. UniInt disconnected startup requires buffering, and it is highly recommended for failover. Buffering for interfaces is configured and enabled through PI ICU.

Buffering services

The PI System offers two services to implement buffering at interfaces:

- PI Buffer Subsystem (PIBufss)
- API Buffer Server (Bufserv)

PI Buffer Subsystem is the best option for most environments.

Use API Buffer Server only if one or more of the following conditions is true:

- The version of PI Data Archive receiving the buffered data is earlier than 3.4.375
- Your interfaces run on a non-Windows platform

If any of the above conditions apply to you, see the *PI Buffering Manager Help (PIPC/HELP/BufferManager.chm)* documentation for PI Server. Otherwise, use PI Buffer Subsystem.

Buffering and collectives

PI Buffer Subsystem 4.3 and later can buffer data to multiple independent servers, including those configured as PI Server collectives. For interfaces to use PI Buffer Subsystem with PI Server collectives, the PI Data Archive servers must be running PI Data Archive version 3.4.375 or later.

Caution:

API Buffer Server does not detect and validate the PI Server collective configuration. As a result, API Buffer Server requires manual configuration whenever a collective changes. Also, because API Buffer Server results in compression at the PI Data Archive machine, archives at different servers in the collective could contain different records.

Buffering configuration

Use PI Interface Configuration Utility (PI ICU) to configure interface buffering.

The **Tools** > **Buffering** option helps you configure buffering. Depending on your current configuration, this option does one of the following:

- If this computer is configured to buffer data using PI Buffer Subsystem 4.3 or later, the Buffering Manager window opens and shows a buffering dashboard. The dashboard shows information about the status of buffering on this computer.
- If this computer is not currently configured to buffer data, and PI Buffer Subsystem 4.3 or later is installed, you are prompted to configure PI Buffer Subsystem. If you click **Yes**, the Buffering Manager window opens and shows the installation wizard, which helps you configure PI Buffer Subsystem.
- If this computer is configured to buffer data using API Buffer Server (Bufserv), and PI Buffer Subsystem 4.3 or later is installed, you are prompted to convert to and configure PI Buffer Subsystem. If you click Yes at both prompts, the Buffering Manager window opens and shows the upgrade wizard, which helps you upgrade from API Buffer Server to PI Buffer Subsystem.
- If PI Buffer Subsystem 4.3 is not yet installed, the Buffering window opens for API Buffering or the earlier version of PI Buffer Subsystem.

For PI Buffer Subsystem 4.3, when configuring an interface to buffer data to a PI Data Archive server which has not been added to the buffered server list you must enable buffering. Click the **Enable** button in the **Buffering Status** box on the interface General page. To verify that the buffering status is `On`, exit PI ICU, then restart and select the interface.

You can use Buffering Manager to configure, monitor, and troubleshoot buffering using PI Buffer Subsystem. PI Buffer Subsystem is recommended for applications connecting to PI Data Archive 3.4.375 or later. Older versions of PI Data Archive require API Buffer Server, as do some sites with custom solutions. See Buffering Manager help *(PIPC/HELP/BufferManager.chm)* for more information.

Chapter 14. Interface Diagnostics Configuration

The PI Point Configuration chapter provides information on building PI points for collecting data from the device. This chapter describes the configuration of points related to interface diagnostics.

Note: The procedure for configuring interface diagnostics is not specific to this interface. Thus, for simplicity, the instructions and screenshots that follow refer to an interface named **ModbusE**.

Some of the points that follow refer to a "performance summary interval". This interval is 8 hours by default. You can change this parameter via the *Scan performance summary* box in the *UniInt – Debug* parameter category page:

Scan Class Performance Points

A scan class performance point measures the amount of time (in seconds) that this interface takes to complete a scan. The interface writes this scan completion time to millisecond resolution. Scan completion times close to 0 indicate that the interface is performing optimally. Conversely, long scan completion times indicate an increased risk of missed or skipped scans. To prevent missed or skipped scans, you should distribute the data collection points among several scan classes.

You configure one scan class performance point for each scan class in this interface. From the ICU, select this interface from the *Interface* drop-down list and click *UniInt-Performance Points* in the parameter category pane:

Right click the row for a particular *Scan Class #* to open the shortcut menu:

You need not restart the interface for it to write values to the scan class performance points.

To see the current values (snapshots) of the scan class performance points, right-click and select *Refresh Snapshots*.

Create / Create All

To create a performance point, right-click the line belonging to the point to be created, and select *Create*. Click *Create All* to create all the scan class performance points.

Delete

To delete a performance point, right-click the line belonging to the point to be deleted and select *Delete*.

Correct / Correct All

If the "Status" of a point is marked "Incorrect", the point configuration can be automatically corrected by the ICU by right-clicking on the line belonging to the point to be corrected and selecting *Correct*. The performance points are created with the following PI attribute values.

Attribute	Details
Tag	Point name that appears in the list box
Point Source	Point Source for points for this interface, as specified on the *General* page
Compressing	Off
Excmax	0
Descriptor	*Interface name* + " Scan Class # Performance Point"

If the ICU detects that a performance point is not defined with the correct attributes, the point will be marked *Incorrect*: To correct all points, click *Correct All*.

Rename

Right-click the line belonging to the point and select *Rename* to rename the performance point.

Column descriptions

Status

The *Status* column in the performance points table indicates whether the performance point exists for the scan class in the *Scan Class #* column.

Created – Indicates that the performance point does exist

Not Created – Indicates that the performance point does not exist

Deleted – Indicates that a performance point existed but was just deleted by the user

Scan Class #

The *Scan Class* column indicates to which scan class the performance point in the *Tagname* column belongs. There will be one scan class in the *Scan Class* column for each scan class listed in the *Scan Classes* box on the *General* page.

Tagname

The *Tagname* column holds the performance point name.

PS

This is the point source used for these performance points and the interface.

Location1

This is the value used by the interface for the **`/ID=`**# point attribute.

ExDesc

This is the used to tell the interface that these are performance points and the value is used to correspond to the **`/ID=`**# command line parameter if multiple copies of the same interface are running on the interface node.

Snapshot

The *Snapshot* column holds the snapshot value of each performance point that exists in the PI Data Archive. The *Snapshot* column is updated when the *Performance Points* page is selected, and when the interface is first loaded. You may have to scroll to the right to see the snapshots.

Performance Counters Points

Installing the interface as a Windows service creates Windows performance counters to report performance data. Such data include items like:

- the amount of time that the interface has been running;
- the number of points the interface has added to its point list;
- the number of points that are currently updating among others

There are two types or instances of performance counters that can be collected and stored in PI points. The first is (_Total) which is a total for the performance counter since the interface instance was started. The other is for individual scan classes (Scan Class *x*) where *x* is a particular scan class defined for the interface instance that is being monitored.

OSIsoft's PI Interface for Performance Monitor is capable of reading these performance values and writing them to PI points. Please see the *PI Interface for Performance Monitor* for more information.

If there is no PI Interface for Performance Monitor registered with the ICU in the Module Database for the PI Data Archive the interface is sending its data to, you cannot use the ICU to create any interface instance's performance counters points:

After installing the PI Interface for Performance Monitor as a service, select this interface instance from the *Interface* drop-down list. Click *Performance Counters* in the parameter categories pane and then right-click on the row containing the performance counter point you wish to create. This will open the shortcut menu:

Click *Create* to create the performance counter point for that particular row. Click *Create All* to create all the performance counters points listed which have a status of Not Created.

To see the current values (snapshots) of the created performance counters points, right-click on any row and select *Refresh Snapshots*.

Note: The PI Interface for Performance Monitor – and not this interface – is responsible for updating the values for the performance counters points in the PI Data Archive. So, make sure that the PI Interface for Performance Monitor is running correctly.

Performance Counters

In the following lists of performance counters, the naming convention is:

“PerformanceCounterName” (.PerformanceCounterPointSuffix)

The name created by the ICU for each performance counter point is based on the setting found under the Tools ➔ Options ➔ Naming Conventions ➔ Performance Counter Points. The default for this is “sy.perf.[machine].[if service]” followed by the performance counter point suffix.

Performance Counters for both (_Total) and (Scan Class x)

“Point Count” (.point_count)

A .sched_scans_%missed performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%missed performance counter point indicates the percentage of scans the interface missed per scan class or the total number missed for all scan classes since startup. A missed scan occurs if the interface performs the scan one second later than scheduled.

The ICU uses a naming convention such that the point containing “(Scan Class 1)” (for example, "“`y.perf.etamp390.E1(Scan Class 1).point_count`"” refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing “(_Total)” refers to the sum of all scan classes.

“Scheduled Scans: % Missed” (.sched_scans_%missed)

A .sched_scans_%skipped performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%skipped performance counter point indicates the percentage of scans the interface skipped per scan class or the total number skipped for all scan classes since startup. A skipped scan is a scan that occurs at least one scan period after its scheduled time. This point is similar to the [UI_SCSKIPPED] health point.

The ICU uses a naming convention such that the point containing "“Scan Class 1)" (for example, "“`y.perf.etamp390.E1(Scan Class 1).sched_scans_%missed`"” refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing "“_Total)"”refers to the sum of all scan classes.

“Scheduled Scans: % Skipped” (.sched_scans_%skipped)

A .sched_scans_%skipped performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_%skipped performance counter point indicates the percentage of scans the interface skipped per scan class or the total number skipped for all scan classes since startup.

A skipped scan is a scan that occurs at least one scan period after its scheduled time. This point is similar to the [UI_SCSKIPPED] health point.

The ICU uses a naming convention such that the point containing "“Scan Class 1)" (for example, "`sy.perf.etamp390.E1(Scan Class 1).sched_scans_%skipped`") refers to scan class 1, "“Scan Class 2)"”refers to scan class 2, and so on. The point containing "“_Total)"”refers to the sum of all scan classes.

“Scheduled Scans: Scan count this interval” (.sched_scans_this_interval)

A *.sched_scans_this*_interval performance counter point is available for each scan class of this interface as well as a "“_Total)"”for the interface instance.

The .sched_scans_this_interval performance counter point indicates the number of scans that the interface performed per performance summary interval for the scan class or the total number of scans performed for all scan classes during the summary interval. This point is similar to the [UI_SCSCANCOUNT] health point.

The ICU uses a naming convention such that the point containing “(Scan Class 1)” (for example, "`sy.perf.etamp390.E1(Scan Class 1).sched_scans_this_interval`") refers to scan class 1, “(Scan Class 2)” refers to scan class 2, and so on. The point containing “(_Total)” refers to the sum of all scan classes.

Performance Counters for (_Total) only

“Device Actual Connections” (.Device_Actual_Connections)

The *.Device_Actual_Connections* performance counter point stores the actual number of foreign devices currently connected and working properly out of the expected number of foreign device connections to the interface. This value will always be less than or equal to the Device Expected Connections counter.

“Device Expected Connections” (.Device_Expected_Connections)

The *.Device_Expected_Connections* performance counter point stores the total number of foreign device connections for the interface. This is the expected number of foreign device connections configured that should be working properly at runtime. If the interface can only communicate with 1 foreign device then the value of this counter will always be one. If the interface can support multiple foreign device connections then this is the total number of expected working connections configured for this interface.

“Device Status” (.Device_Status)

The *.Device_Status* performance counter point stores communication information about the interface and the connection to the foreign device(s). The value of this counter is based on the expected connections, actual connections and value of the **`/PercentUp`** command line option. If the device status is good then the value is ‘0’. If the device status is bad then the value is ‘1’. If the interface only supports connecting to 1 foreign device then the **`/PercentUp`** command line value does not change the results of the calculation. If for example the interface can connect to 10 devices and 5 are currently working then the value of the **`/PercentUp`** command line parameter is applied to determine the Device Status. If the value of the **`/PercentUp`** command line parameter is set to 50 and at least 5 devices are working then the DeviceStatus will remain good (that is, have a value of zero).

"Failover Status" (.Failover_Status)

The *.Failover_Status* performance counter point stores the failover state of the interface when configured for UniInt failover. The value of the counter will be '0' when the interface is running as the primary interface in the failover configuration. If the interface is running in backup mode then the value of the counter will be '1'.

"Interface up-time (seconds)" (.up_time)

The *.up_time* performance counter point indicates the amount of time (in seconds) that this interface has been running. At startup the value of the counter is zero. The value will continue to increment until it reaches the maximum value for an unsigned integer. Once it reaches this value then it will start back over at zero.

"IO Rate (events/second)" (.io_rates)

The *.io_rates* performance counter point indicates the rate (in event per second) at which this interface writes data to its input points.

"Log file message count" (.log_file_msg_count)

The *.log_file_msg_count* performance counter point indicates the number of messages that the interface has written to the log file. This point is similar to the [UI_MSGCOUNT] health point.

"PI Status" (PI_Status)

The *.PI_Status* performance counter point stores communication information about the interface and the connection to the PI Data Archive. If the interface is properly communicating with the PI Data Archive then the value of the counter is '0'. If the communication to the PI Data Archive goes down for any reason then the value of the counter will be '1'. Once the interface is properly communicating with the PI Data Archive again then the value will change back to '0'.

"Points added to the interface" (.pts_added_to_interface)

The *.pts_added_to_interface* performance counter point indicates the number of points the interface has added to its point list. This does not include the number of points configured at startup. This is the number of points added to the interface after the interface has finished a successful startup.

"Points edited in the interface"(.pts_edited_in_interface)

The *.pts_edited_in_interface* performance counter point indicates the number of point edits the interface has detected. The interface detects edits for those points whose PointSource attribute matches the **`/ps=`** parameter and whose Location1 attribute matches the **`/id=`** parameter of the interface.

"Points Good" (.Points_Good)

The *.Points_Good* performance counter point is the number of points that have sent a good current value to the PI Data Archive. A good value is defined as any value that is not a system digital state value. A point can either be Good, In Error, or Stale. The total of Points

Good, Points In Error, and Points State will equal the Point Count. There is one exception to this rule. At startup of an interface, the Stale timeout must elapse before the point will be added to the Stale Counter. Therefore the interface must be up and running for at least 10 minutes for all points to belong to a particular Counter.

"Points In Error" (.Points_In_Error)

The *.Points_In_Error* performance counter point indicates the number of points that have sent a current value to the PI Data Archive that is a system digital state value. Once a point is in the In Error count it will remain in the In Error count until the point receives a new, good value. Points categorized as in Error do not transition to the Stale Counter. Only good points become stale.

"Points removed from the interface" (.pts_removed_from_interface)

The *.pts_removed_from_interface* performance counter point indicates the number of points that have been removed from the interface configuration. A point can be removed from the interface when one of the point attributes is updated and the point is no longer a part of the interface configuration. For example, changing the PointSource, Location1, or Scan attribute can cause the point to no longer be a part of the interface configuration.

"Points Stale 10(min)" (.Points_Stale_10min)

The *.Points_Stale_10min* performance counter point indicates the number of good points that have not received a new value in the last 10 minutes. If a point is Good, then it will remain in the good list until the Stale timeout elapses. At this time if the point has not received a new value within the Stale Period then the point will move from the Good count to the Stale count. Only points that are Good can become Stale. If the point is in the In Error count then it will remain in the In Error count until the error clears. As stated above, the total count of Points Good, Points In Error, and Points Stale will match the Point Count for the interface.

"Points Stale 30(min)" (.Points_Stale_30min)

The *.Points_Stale_30min* performance counter point indicates the number of points that have not received a new value in the last 30 minutes. For a point to be in the Stale 30 minute count it must also be a part of the Stale 10 minute count.

"Points Stale 60(min)" (.Points_Stale_60min)

The *.Points_Stale_60min* performance counter point indicates the number of points that have not received a new value in the last 60 minutes. For a point to be in the Stale 60 minute count it must also be a part of the Stale 10 minute and 30 minute count.

"Points Stale 240(min)" (.Points_Stale_240min)

The *.Points_Stale_240min* performance counter point indicates the number of points that have not received a new value in the last 240 minutes. For a point to be in the Stale 240 minute count it must also be a part of the Stale 10 minute, 30 minute and 60 minute count.

Performance Counters for (Scan Class x) only

"Device Scan Time (milliseconds)" (.Device_Scan_Time)

A *.Device_Scan_Time* performance counter point is available for each scan class of this interface.

The *.Device_Scan_Time* performance counter point indicates the number of milliseconds the interface takes to read the data from the foreign device and package the data to send to the PI Data Archive. This counter does not include the amount of time to send the data to the Pi Data Archive. This point is similar to the [UI_SCINDEVSCANTIME] health point.

The ICU uses a naming convention such that the point containing "(Scan Class 1)" (for example, "`sy.perf.etamp390.E1 (Scan Class 1).device_scan _time`") refers to scan class 1, "(Scan Class 2) refers to scan class 2, and so on.

"Scan Time (milliseconds)" (.scan_time)

A *.scan_time* performance counter point is available for each scan class of this interface.

The *.scan_time* performance counter point indicates the number of milliseconds the interface takes to both read the data from the device and send the data to the PI Data Archive. This point is similar to the [UI_SCINSCANTIME] health point.

The ICU uses a naming convention such that the point containing "(Scan Class 1)" (for example, "`sy.perf.etamp390.E1(Scan Class 1).scan_time"`) refers to scan class 1, "(Scan Class 2)" refers to scan class 2, and so on.

Interface Health Monitoring Points

Interface Health Monitoring Points provide information about the health of this interface. To use the ICU to configure these points, select this interface from the *Interface* drop-down list and click *Health Points* from the parameter category pane:

Right click the row for a particular health point to display the shortcut menu:

Click *Create* to create the health point for that particular row. Click *Create All* to create all the health points.

To see the current values (snapshots) of the health points, right-click and select *Refresh Snapshots*.

For some of the health points described subsequently, the interface updates their values at each performance summary interval (typically, 8 hours).

[UI_HEARTBEAT]

The [UI_HEARTBEAT] health point indicates whether the interface is currently running. The value of this point is an integer that increments continuously from 1 to 15. After reaching 15, the value resets to 1.

The fastest scan class frequency determines the frequency at which the interface updates this point:

Fastest Scan Frequency	Update frequency
Less than 1 second	1 second
Between 1 and 60 seconds, inclusive	Scan frequency
More than 60 seconds	60 seconds

If the value of the [UI_HEARTBEAT] health point is not changing, then this interface is in an unresponsive state.

[UI_DEVSTAT]

The [UI_DEVSTAT] health point provides an indication of the connection status between the interface and the source device. The possible values for this string point are:

- "1 | Starting | UI #.#.#.#" – The interface is currently starting up.
- "Good" – The interface is properly communicating and reading data from all configured PMUs. A value of "Good" does not mean that all points are receiving good values, but it is a good indication that there are no hardware or network problems.
- "3 | 1 devices(s) in error | REP: 1 LEP 1" – The connection to one or more PMUs has failed.
- "4 | Intf Shutdown" – The interface has been shutdown.

Refer to the *PI Universal Interface (UniInt) User Guide*.doc file for more information on how to configure health points.

[UI_SCINFO]

The [UI_SCINFO] health point provides scan class information. The value of this point is a string that indicates

- the number of scan classes;
- the update frequency of the [UI_HEARTBEAT] health point ; and
- the scan class frequencies

An example value for the [UI_SCINFO] health point is:

```
3 | 5 | 5 | 60 | 120
```

The interface updates the value of this point at startup and at each performance summary interval.

[UI_IORATE]

The [UI_IORATE] health point indicates the sum of

1. the number of scan-based input values the interface collects before it performs exception reporting; and
2. the number of event-based input values the interface collects before it performs exception reporting

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The value of this [UI_IORATE] health point may be zero. A stale timestamp for this point indicates that this interface has stopped collecting data.

[UI_MSGCOUNT]

The [UI_MSGCOUNT] health point tracks the number of messages that the interface has written to the log file since start-up. In general, a large number for this point indicates that the interface is encountering problems. You should investigate the cause of these problems by looking in log messages.

The interface updates the value of this point every 60 seconds. While the interface is running, the value of this point never decreases.

[UI_POINTCOUNT]

The [UI_POINTCOUNT] health point counts number of PI points loaded by the interface. This count includes all input and triggered input points. This count does NOT include any interface health points or performance points.

The interface updates the value of this point at startup, on change and at shutdown.

[UI_TRIGGERRATE]

The [UI_TRIGGERRATE] health point tracks the number of values that the interface writes to event-based input points. If there are no event-based input points for this interface, it writes the System Digital State `No Result` to this health point.

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The interface resets the value of this point to zero at each performance summary interval.

[UI_TRIGGERBVRATE]

The [UI_TRIGGERBVRATE] health point tracks the number of system digital state values that the interface writes to event-based input points. If there are no event-based input points for this interface, it writes the system digital state `No Result` to this health point.

The interface updates this point at the same frequency as the [UI_HEARTBEAT] point. The interface resets the value of this point to zero at each performance summary interval.

[UI_SCIORATE]

You can create a [UI_SCIORATE] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class IO Rate.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCIORATE] health point indicates the number of values that the interface has collected. If the current value of this point is between zero and the corresponding [UI_SCPOINTCOUNT] point, inclusive, then the interface executed the scan successfully. If a [UI_SCIORATE] point stops updating, then this condition indicates that an error has occurred and the points in the scan class are no longer receiving new data.

The interface updates the value of a [UI_SCIORATE] point after the completion of the associated scan.

Although the ICU allows you to create the point with the suffix ".sc0", this point is not applicable to this interface.

[UI_SCBVRATE]

You can create a [UI_SCBVRATE] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Bad Value Rate.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCBVRATE] point indicates the number System Digital State values that the interface has collected.

The interface updates the value of a [UI_SCBVRATE] point after the completion of the associated scan.

Although the ICU allows you to create the point with the suffix ".sc0", this point is not applicable to this interface.

[UI_SCSCANCOUNT]

You can create a [UI_SCSCANCOUNT] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Scan Count.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_ SCSCANCOUNT] point tracks the number of scans that the interface has performed.

The interface updates the value of this point at the completion of the associated scan. The interface resets the value to zero at each performance summary interval.

Although there is no "Scan Class 0", the ICU allows you to create the point with the suffix ".sc0". This point indicates the total number of scans the interface has performed for all of its Scan Classes.

[UI_SCSKIPPED]

You can create a [UI_SCSKIPPED] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix ".sc1" (for example, `sy.st.etamp390.E1.Scan Class Scans Skipped.sc1`) refers to scan class 1, ".sc2" refers to scan class 2, and so on.

A particular scan class's [UI_SCSKIPPED] point tracks the number of scans that the interface was not able to perform before the scan time elapsed and before the interface performed the next scheduled scan.

The interface updates the value of this point each time it skips a scan. The value represents the total number of skipped scans since the previous performance summary interval. The interface resets the value of this point to zero at each performance summary interval.

Although there is no “Scan Class 0”, the ICU allows you to create the point with the suffix “.sc0”. This point monitors the total skipped scans for all of the interface’s Scan Classes.

[UI_SCPOINTCOUNT]

You can create a [UI_SCPOINTCOUNT] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Point Count.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

This health point monitors the number of points in a scan class.

The interface updates a [UI_SCPOINTCOUNT] health point when it performs the associated scan.

Although the ICU allows you to create the point with the suffix “.sc0”, this point is not applicable to this interface.

[UI_SCINSCANTIME]

You can create a [UI_SCINSCANTIME] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Scan Time.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

A particular scan class’’ [UI_ SCINSCANTIME] point represents the amount of time (in milliseconds) the interface takes to read data from the device, fill in the values for the points, and send the values to the PI Data Archive.

The interface updates the value of this point at the completion of the associated scan.

[UI_SCINDEVSCANTIME]

You can create a [UI_SCINDEVSCANTIME] health point for each scan class in this interface. The ICU uses a point naming convention such that the suffix "“sc1"”(for example, `sy.st.etamp390.E1.Scan Class Device Scan Time.sc1`) refers to scan class 1, "“sc2"”refers to scan class 2, and so on.

A particular scan class'' [UI_ SCINDEVSCANTIME] point represents the amount of time (in milliseconds) the interface takes to read data from the device and fill in the values for the points.

The value of a [UI_ SCINDEVSCANTIME] point is a fraction of the corresponding [UI_SCINSCANTIME] point value. You can use these numbers to determine the percentage of time the interface spends communicating with the device compared with the percentage of time communicating with the PI Data Archive.

If the [UI_SCSKIPPED] value is increasing, the [UI_SCINDEVSCANTIME] points along with the [UI_SCINSCANTIME] points can help identify where the delay is occurring: whether the reason is communication with the device, communication with the PI Data Archive, or elsewhere.

The interface updates the value of this point at the completion of the associated scan.

I/O Rate Point

An I/O Rate point measures the rate at which the interface writes data to its input points. The value of an I/O Rate point represents a 10-minute average of the total number of values per minute that the interface sends to the PI Data Archive.

When the interface starts, it writes 0 to the I/O Rate point. After running for ten minutes, the interface writes the I/O Rate value. The interface continues to write a value every 10 minutes. When the interface stops, it writes 0.

The ICU allows you to create one I/O Rate point for each copy of this interface. Select this interface from the *Interface* drop-down list, click *IO Rate* in the parameter category pane, and check *Enable IORates for this interface*.

As the preceding picture shows, the ICU suggests an *Event Counter* number and a *Tagname* for the I/O Rate point. Click the *Save* button to save the settings and create the I/O Rate point. Click the *Apply* button to apply the changes to this copy of the interface.

You need to restart the interface in order for it to write a value to the newly created I/O Rate point. Restart the interface by clicking the *Restart* button:

(The reason you need to restart the interface is that the PointSource attribute of an I/O Rate point is `Lab`.)

To confirm that the interface recognizes the I/O Rate point, look in the log file for a message such as:

```
PI-ModBus 1> IORATE: tag sy.io.etamp390.ModbusE1 configured.
```

To see the I/O Rate point's current value (snapshot), click the *Refresh* button of the *Snapshot* box:

Enable IORates for this Interface

The *Enable IORates for this interface* check box enables or disables I/O Rates for the current interface. To disable I/O Rates for the selected interface, uncheck this box. To enable I/O Rates for the selected interface, check this box.

Event Counter

The *Event Counter* correlates a point specified in the iorates.dat file with this copy of the interface. The command-line equivalent is **`/ec=`***x*, where *x* is the same number that is assigned to a point name in the iorates.dat file.

Tagname

The name listed in the *Tagname* box is the name of the I/O Rate point.

Tag Status

The *Tag Status* box indicates whether the I/O Rate point exists in the PI Data Archive. The possible states are:

- Created – This status indicates that the point exist in the PI Data Archive
- Not Created – This status indicates that the point does not yet exist in the PI Data Archive
- Deleted – This status indicates that the point has just been deleted
- Unknown – This status indicates that the PI ICU is not able to access the PI Data Archive

In File

The *In File* box indicates whether the I/O Rate point listed in the *Tagname* box and the number in the *Event Counter* is in the IORates.dat file. The possible states are:

- Yes – This status indicates that the point name and event counter are in the IORates.dat file
- No – This status indicates that the point name and event counter are not in the IORates.dat file

Snapshot

The *Snapshot* column holds the snapshot value of the I/O Rate point, if the I/O Rate point exists in the PI Data Archive. The *Snapshot* box is updated when the *IORate* page is selected, and when the interface is first loaded.

Create/Save

Create the suggested I/O Rate point with the name indicated in the Tagname box. Or, save any changes for the point name indicated in the Tagname box.

Delete

Delete the I/O Rate point listed in the *Tagname* box.

Rename

Change the name for the I/O Rate point listed in the *Tagname* box.

Add to File

Add the point indicated by the name in the Tagname box to the IORates.dat file with the event counter listed in the *Event Counter* box.

Search

Search the PI Data Archive for a previously defined I/O Rate point.

Appendix A. Error and Informational Messages

A string `NameID` is pre-pended to error messages written to the message log. `Name` is a non-configurable identifier that is no longer than 9 characters. *ID* is a configurable identifier that is no longer than 9 characters and is specified using the **`/id`** parameter on the startup command-line.

Message Logs

The location of the message log depends upon the platform on which the interface is running. See the *PI Universal Interface (UniInt) User Guide* for more information.

For more information about logs for interfaces running on Windows, see *UniInt Interface Message Logging for UniInt 4.5.0.x and later Interfaces* or OISsoft KB article - KB00401 - How to read new UniInt 4.5.0.x and later Interface message logs on the OSIsoft technical support web site.

Messages are written to `[PIHOME]\dat\pipc.log` at the following times.

- When the interface starts many informational messages are written to the log. These include the version of the interface, the version of UniInt, the command-line parameters used, and the number of points.
- As the interface loads points, messages are sent to the log if there are any problems with the configuration of the points.
- If the UniInt **`/dbUniInt`** parameter is found in the command-line, then various informational messages are written to the log file.

Messages

Interface Configuration Error Messages

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Schema instance, make sure MSXML 4.0 is installed.`
	Or
	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Document instance, make sure MSXML 4.0 is installed.`
	Or
	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't create XML DOM Schema Collection object, make sure MSXML 4.0 is installed.`

Cause	There was a problem created one of the DOM XML objects. Typically indicates that the Microsoft MSXML 4.0 API is not installed.
Resolution	Download the API from Microsoft and install.

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig – XML config document load failed: System error: -2146697210.`
Cause	The specified XML file could not be loaded.
Resolution	Check the file name and path specified via the /confile argument.

Message	`ERROR> Hardware initialization – C37118XMLConfig::C37118XMLConfig - Couldn't find root <PI_C37_118.Session> element in the XML configuration document.`
Cause	The root node root <PI_C37_118.Session> couldn't be located.
Resolution	Check the format of the XML file. If using the PIC37118 ICU (recommended) this should be correct.

Message	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for DataSortingAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for SyncErrorAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for PMUErrorActionmust be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for DataSortingAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for LeapSecondAction must be 0, 1, 2 or 3.`
	Or
	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - Value for WriteIOTimeout must be 0 or 1.`

Cause	The value specified for the one of the above attributes in the root section of the XML configuration file is out of range.
Resolution	Use PIC37118 ICU to ensure this value is correct.

Message	`ERROR> Hardware initialization - C37118XMLConfig::C37118XMLConfig - No <LocalEndPoint> elements found in the PIC37118_Arbiter_Example.xml XML configuration document.`
Cause	No <LocalEndPoints> section was found in the configuration document.
Resolution	The must be at least one LocalEndPoints section in the file.
Message	`LocalEndPoint::LocalEndPoint( -1 ) - ID attribute missing from LocalEndPoint definition.`
Cause	The required ID attribute is missing from a LocalEndPoint section.
Resolution	Add the ID attribute to the LocalEndPoint definition.

Message	`LocalEndPoint::LocalEndPoint( 2 ) - Type element in LocalEndpoint configuration must be IP or Serial.`
Cause	The Type attribute in LocalEndPoint with ID of 2 has invalid communications type specified.
Resolution	Use the PIC37118 ICU to ensure this attribute is correct.

Message	`LocalEndPoint::LocalEndPoint( 2 ) - Invalid data protocol specified in <DataProtocol>, must be TCP or UDP(case insensitive).`
	Or
	`LocalEndPoint::LocalEndPoint( 2 ) - Invalid data protocol specified in <ConfigProtocol>, must be TCP or UDP(case insensitive).`

Cause	The DataProtocol or ConfigProtocol attribute must be TCP or UDP.
Resolution	Use the PIC37118 ICU to ensure this attribute is correct. This parameter is not case sensitive.

Point Loading Error Messages

The following messages are logged when there is a problem loading a point. In each case the point name and point id is reported along with the error information.

Message	`LocalEndPoint Id: 1 not found in interface LocalEndPointCollection`
Cause	There is not a LocalEndPoint defined in the XML configuration file that matches the Location2 attribute of the point.
Resolution	Add the appropriate LocalEndPoint.

Message	`RemoteEndPoint Id: 1 not found in interface RemoteEndPointCollection`
Cause	There is not a RemoteEndPoint defined in the XML configuration file that matches the Location3 attribute of the point.
Resolution	Add the appropriate RemoteEndPoint

Message	`Delimiter character '\' not found in InstrumentTag specification (PMU.FREQ)`
Cause	The InstrumentTag attribute requires the Measurement Type and Measurement be delimited by a "/" character and was not found.
Resolution	Added appropriate delimiter.
Message	`RemoteEndPoint tag of type LEAPSEC is already defined`
Cause	There can only be a single point for a given RemoteEndPoint type(REP) for each RemoteEndPoint. For example there cannot be two points with an InstrumentTag field of REP/LEAPSEC configured for the same RemoteEndPoint. Any subsequent points are rejected.
Resolution	Remove duplicate point.

Message	`PMU Id: 66 not found in interface PMU Collection`
Cause	The PMU ID specified in the Location5 attribute was not found in the XML configuration file.
Resolution	Fix XML configuration file to include ID.

Message	`Phasor measurement string XX is shorter then shortest valid keyword`
	Or
	`Invalid Phasor sub type ( XXX ) specified`

Cause	The Phasor measurement specified in the InstrumentTag attribute is invalid. This value must be PhasorReal, PhasorImag, PhasorMag or PhasorAngle.
Resolution	Correct XML configuration file.

Message	`Phasor channel ( XXX ) is not configured`
	`Or`
	`Analog channel ( XXX ) is not configured`
	`Or`
	`Digital channel ( XXX ) is not configured`

Cause	The Phasor, analog or digital channel name is not valid for the PMU. This value is not case sensitive but imbedded spaces must remain if they are present in the channel names reported by the PMU. Trailing spaces may be removed.
Resolution	Correct XML configuration file.

Message	`Phasor tag with channel name of XXX and sub-type of IMAG is already defined for PMU`
	Or
	`Analog tag with channel name of XXX is already defined for PMU`
	Or
	`Digital tag with channel name of XXX is already defined for PMU`

Cause	There can only be one PI point configured for each Analog or Digital channel. Each Phasor channel can have one PI point for each sub type on a channel.
Resolution	Correct by removing the duplicate point entry.

Message	`Invalid PMU measurement type ( XXX ) specified`
Cause	The PMU measurement specified in the InstrumentTag is not valid. For example: PMU/XXX is not valid but PMU/FREQ is.
Resolution	Correct the invalid measurement type.

Communications Related Error Messages

The following messages are logged when there is a problem communications to the C37.118 device. In each case the communications parameters are reported along with the error information.

Message	`PDC/PMU communication timeout.`
Cause	The communications with a PDC or PMU have timed out. The interface will continue to attempt a reconnection.

Message	`Error code: -1, Generic Error`
Cause	The interface was unable to determine the exact nature of the error. Any available information is displayed.

Message	`Debug code: -2, Debug Message.`
Cause	Miscellaneous debug message.

Message	`Error code: -3, Truncated String.`
Cause	The C37.118 message was truncated.

Message	`Error code: -4, Failed to create communication instance.`
Cause	The interface was unable to create a communications thread. Check system resources.
Resolution	Check system resources.

Message	`Error code: -5, Bad memory allocation.`
Cause	The UFO_ID pa
Resolution	Check system resources.

Message	`Error code: -6, Required parameter passed a NULL pointer.`
Cause	An internal function passed a NULL pointer for a required variable. This should never occur.
Resolution	Contact OSIsoft technical support.

Message	`Error code: -7, Failed to stop communication thread.`
Cause	The communications thread could not be stopped. This would only occur during interface shutdown, but should never occur.
Resolution	Contact OSIsoft technical support.

Message	`Error code: -8, PDC/PMU communication timeout.`
Cause	The communications with a PDC or PMU have timed out. The interface will continue to attempt a reconnection.

Message	`Error code: -9, Sync byte not found.`
Cause	The 0xAA sync byte was not found in the data stream. The interface will attempt to resynchronize the data stream by closing the socket and reconnecting.
Message	`Error code: -10, Message IDCODE not for PDC / PMU.`
Cause	The message received was valid but the IDCODE in word 3 of the frame was not for the configured PDC / PMU.
Resolution	Check the XML configuration file and PDC / PMU configuration.

Message	`Error code: -11, Message frame size error.`
Cause	The frame size in the C37.118 message did not match the actual size.

Message	`Error code: -12, Message CRC error.`
Cause	The CRC validation failed on a C37.118 message.
Resolution	This error may occur occasionally. If it occurs with any frequency, check the communications hardware.

Message	`Error code: -13, No Message Buffer Available.`
Cause	No message buffer was available. The interface will continue to operate but will be missing data.
Resolution	Contact OSIsoft technical support.

System Errors and PI Errors

System errors are associated with positive error numbers. Errors related to PI are associated with negative error numbers.

Error Descriptions

Descriptions of system and PI errors can be obtained with the pidiag utility:

```
\PI\adm\pidiag /e error_number
```

UniInt Failover Specific Error Messages

Informational

Message	`16-May-06 10:38:00` `C37118 1> UniInt failover: Interface in the “Backup” state.`
Meaning	Upon system startup, the initial transition is made to this state. While in this state the interface monitors the status of the other interface participating in failover. When configured for Hot failover, data received from the data source is queued and not sent to the PI Data Archive while in this state. The amount of data queued while in this state is determined by the failover update interval. In any case, there will be typically no more than two update intervals of data in the queue at any given time. Some transition chains may cause the queue to hold up to five failover update intervals worth of data.

Message	`16-May-06 10:38:05` `C37118 1> UniInt failover: Interface in the “Primary” state and actively sending data to PI. Backup interface not available.`
Meaning	While in this state, the interface is in its primary role and sends data to the PI Data Archive as it is received. This message also states that there is not a backup interface participating in failover.

Message	`16-May-06 16:37:21` `C37118 1> UniInt failover: Interface in the “Primary” state and actively sending data to PI. Backup interface available.`
Meaning	While in this state, the interface sends data to the PI Data Archive as it is received. This message also states that the other copy of the interface appears to be ready to take over the role of primary.

Errors (Phase 1 & 2)

Message	`16-May-06 17:29:06`
	`C37118 1> One of the required Failover Synchronization points was not loaded.`
	`Error = 0: The Active ID synchronization point was not loaded.`
	`The input PI tag was not loaded`

Cause	The active ID point is not configured properly.
Resolution	Check validity of point attributes. For example, make sure Location1 attribute is valid for the interface. All failover tags must have the same `PointSource` and `Location1` attributes. Modify point attributes as necessary and restart the interface.

Message	`16-May-06 17:38:06`
	`C37118 1> One of the required Failover Synchronization points was not loaded.`
	`Error = 0: The Heartbeat point for this copy of the interface was not loaded.`
	`The input PI tag was not loaded`

Cause	The heartbeat point is not configured properly.
Resolution	Check validity of point attributes. For example, make sure Location1 attribute is valid for the interface. All failover tags must have the same PointSource and Location1 attributes. Modify point attributes as necessary and restart the interface.

Message	`17-May-06 09:06:03` `C37118 1> The Uniint FailOver ID (`**`/UFO_ID`**`) must be a positive integer.`
Cause	The UFO_ID parameter has not been assigned a positive integer value.
Resolution	Change and verify the parameter to a positive integer and restart the interface.

Message	`17-May-06 09:06:03` `C37118 1> The Failover ID parameter (`**`/UFO_ID`**`) was found but the ID for the redundant copy was not found`
Cause	The **`/UFO_OtherID`** parameter is not defined or has not been assigned a positive integer value.
Resolution	Change and verify the **`/UFO_OtherID`** parameter to a positive integer and restart the interface.

Errors (Phase 2)

Unable to open synchronization file

Message	27-Jun-08 17:27:17 PI Eight Track 1 1> Error 5: Unable to create file '\\georgiaking\GeorgiaKingStorage\UnIntFailover\\PIEightTrack_eight_1.dat' Verify that interface has read/write/create access on file server machine. Initializing UniInt library failed Stopping Interface
Cause	This message will be seen when the interface is unable to create a new failover synchronization file at startup. The creation of the file only takes place the first time either copy of the interface is started and the file does not exist. The error number most commonly seen is error number 5. Error number 5 is an “access denied” error and is likely the result of a permissions problem.
Resolution	Ensure the account the interface is running under has read and write permissions for the folder. The “log on as” property of the Windows service may need to be set to an account that has permissions for the folder.

Error Opening Synchronization File

Message	Sun Jun 29 17:18:51 2008 PI Eight Track 1 2> WARNING> Failover Warning: Error = 64 Unable to open Failover Control File '\\georgiaking\GeorgiaKingStorage\Eight\PIEightTrack_eight_1.dat' The interface will not be able to change state if PI is not available
Cause	This message will be seen when the interface is unable to open the failover synchronization file. The interface failover will continue to operate correctly as long as communication to the PI Data Archive is not interrupted. If communication to the PI Data Archive is interrupted while one or both interfaces cannot access the synchronization file, the interfaces will remain in the state they were in at the time of the second failure, so the primary interface will remain primary and the backup interface will remain backup.
Resolution	Ensure the account the interface is running under has read and write permissions for the folder and file. The “log on as” property of the Windows service may need to be set to an account that has permissions for the folder and file.

Appendix B. PI SDK Options

To access the PI SDK settings for this interface, select this interface from the *Interface* drop-down list and click *UniInt – PI SDK* in the parameter category pane.

Disable PI SDK

Select *Disable PI SDK* to tell the interface not to use the PI SDK. If you want to run the interface in disconnected startup mode, you must choose this option.

The command line equivalent for this option is **`/pisdk=0`**.

Use the Interface's default setting

This selection has no effect on whether the interface uses the PI SDK. However, you must not choose this option if you want to run the interface in disconnected startup mode.

Enable PI SDK

Select *Enable PI SDK* to tell the interface to use the PI SDK. Choose this option if the PI Data Archive version is earlier than 3.4.370.x or the PI API is earlier than 1.6.0.2, and you want to use extended lengths for the Tag, Descriptor, ExDesc, InstrumentTag, or PointSource point attributes. The maximum lengths for these attributes are:

Attribute	Enable the Interface to use the PI SDK	PI Data Archive earlier than 3.4.370.x or PI API earlier than 1.6.0.2, without the use of the PI SDK
Tag	1023	255
Descriptor	1023	26
ExDesc	1023	80
InstrumentTag	1023	32
PointSource	1023	1

However, if you want to run the interface in disconnected startup mode, you must not choose this option.

The command line equivalent for this option is **`/pisdk=1`**.

Appendix C. Communication Plug-In Selection

This appendix describes the selection criteria for the Communications plug-in for the PDCs and PMUs which have been tested. Current Plug-In selections are PIC37118_Comm001, PIC37118_Comm002 and PIC37118_Comm003.

PDC/PMU	Data Protocol	Command Protocol	Command and Data on Same Port	Data Transmission Mode	Close Cmd Comm Setting	Plug-In
Arbiter 1133	UDP	UDP	Yes	Unicast	Yes	001
Arbiter 1133	TCP	TCP	Yes	Unicast	Yes	001
Arbiter 1133	TCP	UDP	No	Unicast	Yes	002
Arbiter 1133	TCP	UDP	No	Broadcast	Yes	002
Arbiter 1133	TCP	UDP	No	Multicast	Yes	002
Areva 897	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
Electric Power Group PDC	UDP	TCP	No	Unicast	Yes	002
NI Rio	TCP	TCP	Yes	Unicast	Yes	001
Siemens R-PMU	TCP	TCP	Yes	Unicast	Yes	001
SEL 3306	TCP	UDP	No	Unicast	No	002
SEL 3378 (SVP operating in PDC mode)	TCP	TCP	Yes	Unicast	Yes	001
SEL 421	UDP	UDP	Yes	Unicast	Yes	001
SEL 421	TCP	TCP	Yes	Unicast	Yes	001
SEL 421	TCP	UDP	No	Unicast	Yes	002
SEL 421	TCP	UDP	No	Broadcast	Yes	002
SEL 421	TCP	UDP	No	Multicast	Yes	002
SEL 451	UDP	UDP	Yes	Unicast	Yes	001
SEL 451	TCP	TCP	Yes	Unicast	Yes	001
SEL 451	TCP	UDP	No	Unicast	Yes	002
SEL 451	TCP	UDP	No	Broadcast	Yes	002
SEL 451	TCP	UDP	No	Multicast	Yes	002
SEL 487E	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
GE D60	TCP	TCP	Yes	Unicast	Yes	001
Siemens R-PMU	TCP	TCP	Yes	Unicast	Yes	001
TVA Proxy	Udp	n/a	n/a	Unicast	No	003
Serial (All Devices)	n/a	n/a	n/a	n/a	No	001

Appendix D. Terminology

To understand this interface manual, you should be familiar with the terminology used in this document.

Interface Specific Terms

IEEE

IEEE stands for the Institute of Electrical & Electronic Engineers.

PMU

A Phasor Measurement Unit (PMU) is a generic device that produces synchronized phasors from voltage and/or current inputs and synchronizing signal.

Phasor

A Phasor is the complex equivalent of a simple sine wave quantity such that the complex modulus is the sine wave amplitude and complex angle (in polar form) is the sine wave phase angle.

PDC

A Phasor Data Concentrator (PDC) provides Phasor data from multiple PMUs.

General Terms

Buffering

Buffering refers to an interface node's ability to store temporarily the data that interfaces collect and to forward these data to the appropriate PI Data Archives.

N-Way Buffering

If you have PI Data Archives that are part of a collective, PIBufss supports n-way buffering. N-way buffering refers to the ability of a buffering application to send the same data to each of the PI Data Archives in a collective. (Bufserv also supports n-way buffering to multiple PI Data Archives however it does not guarantee identical archive records since point compressions attributes could be different between PI Data Archives. With this in mind, OSIsoft recommends that you run PIBufss instead.)

ICU

ICU refers to the PI Interface Configuration Utility. The ICU is the primary application that you use to configure PI interface programs. You must install the ICU on the same computer on which an interface runs. A single copy of the ICU manages all of the interfaces on a particular computer.

You can configure an interface by editing a startup command file. However, OSIsoft discourages this approach. Instead, OSIsoft strongly recommends that you use the ICU for interface management tasks.

ICU Control

An ICU Control is a plug-in to the ICU. Whereas the ICU handles functionality common to all interfaces, an ICU Control implements interface-specific behavior. Most PI interfaces have an associated ICU Control.

Interface Node

An interface node is a computer on which

- the PI API and/or PI SDK are installed, and
- PI Data Archive programs are not installed.

PI API

The PI API is a library of functions that allow applications to communicate and exchange data with the PI Data Archive. All PI interfaces use the PI API.

PI Data Archive Collective

A PI Data Archive Collective is two or more replicated PI Data Archives that collect data concurrently. Collectives are part of the High Availability environment. When the primary PI Data Archive in a collective becomes unavailable, a secondary collective member node seamlessly continues to collect and provide data access to your PI clients.

PI Data Archive Node

A PI Data Archive node is a computer on which PI Data Archive programs are installed. The PI Data Archive runs on the PI Data Archive node. In earlier documentation, PI Data Archive was referred to as the PI Server. (See PI Server Node.)

PIHOME

PIHOME refers to the directory that is the common location for PI 32-bit client applications.

A typical *PIHOME* on a 32-bit operating system is `C:\Program Files\PIPC`.

A typical *PIHOME* on a 64-bit operating system is `C:\Program Files (x86)\PIPC`.

PI 32-bit interfaces reside in a subdirectory of the `Interfaces` directory under `PIHOME`.

For example, files for the 32-bit Modbus Ethernet Interface are in

`[PIHOME]\PIPC\Interfaces\ModbusE`.

This document uses `[PIHOME]` as an abbreviation for the complete `PIHOME` or `PIHOME64` directory path. For example, ICU files in `[PIHOME]\ICU`.

PIHOME64

PIHOME64 is found only on a 64-bit operating system and refers to the directory that is the common location for PI 64-bit client applications.

A typical *PIHOME64* is `C:\Program Files\PIPC`.

PI 64-bit interfaces reside in a subdirectory of the `Interfaces` directory under *PIHOME64*.

For example, files for a 64-bit Modbus Ethernet Interface would be found in

```
C:\Program Files\PIPC\Interfaces\ModbusE.
```

This document uses `[PIHOME]` as an abbreviation for the complete `PIHOME` or *PIHOME64* directory path. For example, ICU files in `[PIHOME]\ICU`.

PI Message Log

The PI message log is the file to which OSIsoft interfaces based on UniInt 4.5.0.x and later write informational, debug and error messages. When a PI interface runs, it writes to the local PI message log. This message file can only be viewed using the PIGetMsg utility. See the *UniInt Interface Message Logging.docx* file for more information on how to access these messages.

PI SDK

The PI SDK is a library of functions that allow applications to communicate and exchange data with the PI Data Archive. Some PI interfaces, in addition to using the PI API, require the use of the PI SDK.

PI Server Node

In earlier documentation, the term "PI Server" was used as a nickname for the PI Data Archive and a PI Server node was a computer on which PI Data Archive programs were installed. While the PI Data Archive remains a core server of the PI Server product, the product name "PI Server" now refers to much more than the PI Data Archive. OSIsoft documentation, including this user manual, is changing to use "PI Server" in this broader sense and "PI Data Archive" to refer to the historian core. (See PI Data Archive Node.)

PI SMT

PI SMT refers to PI System Management Tools. PI SMT is the program that you use for configuring PI Data Archives. A single copy of PI SMT manages multiple PI Data Archives. PI SMT runs on either a PI Data Archive node or an interface node.

Pipc.log

The `pipc.log` file is the file to which OSIsoft interfaces based on UniInt versions earlier than 4.5.0.x write informational and error messages. When a PI interface runs, it writes to the `pipc.log` file. The ICU allows easy access to the `pipc.log`.

Point

The PI point is the basic building block for controlling data flow to and from the PI Data Archive. For a given timestamp, a PI point holds a single value.

A PI point does not necessarily correspond to a "point" on the data source device. For example, a single "point" on the data source device can consist of a set point, a process value,

an alarm limit, and a discrete value. These four pieces of information require four separate PI points.

Service

A Service is a Windows program that runs without user interaction. A service continues to run after you have logged off from Windows. It has the ability to start up when the computer itself starts up.

The ICU allows you to configure a PI interface to run as a service.

Tag

The Tag attribute of a PI point is the name of the PI point. There is a one-to-one correspondence between the name of a point and the point itself. Because of this relationship, PI System documentation uses the terms “tag” and “point” interchangeably.

Interfaces read values from a device and write these values to an input tag.

Appendix E. Technical Support and Resources

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or techsupport@osisoft.com. The OSIsoft Technical Support website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

To ask questions of others who use OSIsoft software, join the OSIsoft user community, PI Square. Members of the community can request advice and share ideas about the PI System. The PI Developers Club space within PI Square offers resources to help you with the programming and integration of OSIsoft products.

Appendix F. Revision History

Date	Author	Comments
13-Jun-2007	Jhartman	Initial Release
18-Jul-2007	Jloe	Version 1.0.1.3 Revision H: updated headers, table formatting, IORates information
26-Jul-2007	Mkelly	Version 1.0.1.3 Revision I: Updated the ICU Control and XML section. Fixed headers and footers. Corrected filenames references for csv files included with distribution.
28-Jun-2008	Jhartman	Version 1.0.2.0, Added /TagConfig argument information and updated version.
10-Oct-2008	Mkelly	Version 1.0.2.0 Revision A; Updated screenshots, replace buffering section, fixed headers and links, updated TOC.
14-Jan-2009	Jhartman	Updated version to 1.0.3.0
02-Feb-2009	Mkelly	Version 1.0.3.0, Revision A; Updated copyright date, added Terminology section, updated TOC.
07-Apr-2009	Jhartman	Changed ranges for EndPoint and PMU Id's to 0-65535 to reflect unsigned short value.
11-Oct-2010	Sbondar	Updated with skeleton Version 3.0.27. Updated the list of tested PMU/PDC
17-Dec-2010	Sbondar	Updated with PDC / PMU Data Source Level Failover and some other additional parameters.
29-Dec-2010	Sbranscomb	Version 1.0.4.x Revision A; Updated with skeleton Version 3.0.31.
28-Jan-2011	Mkelly	Version 1.0.4.x Revision B; Update ICU Control screenshots for new parameters.
10-Feb-2011	Sbondar	Version 1.0.4.x Revision C; Update tables of Attributes and Elements with the new parameters.
15-Feb-2011	Sbondar	Version 1.0.4.x Revision D; Change phrase from 'server-level failover' to 'Data Source Level Failover' to be consistent with other documents. The reason for that is PMU / PDC are considered to be more like a data source rather than a server.
21-Jul-2011	Twilliams	Version 1.0.4.x – 1.0.5.x; Updated the version number for a rebuild with new UniInt 4.5.2.0.
21-Feb-2013	Mkelly	Version 1.0.6.x Updated ICU Control screenshots for new MulticastSourceIPAddress XML Element.
25-Feb-2013	Sbranscomb	Version 1.0.6.x; Revision A; Updated with skeleton Version 3.0.36.
28-Feb-2013	Sbondar	Version 1.0.6.x Revision B; Updated description for new MulticastSourceIPAddress XML Element

Date	Author	Comments
01-Apr-2013	Sbondar	Version 1.0.6.x Revision C; Updated /TagConfig and /configfile parameters in command-line. Parameters table
01-Apr-2013	Sbondar	Version 1.0.6.x Revision D; Updated InstrumentTag attribute description.
01-July-2013	Sbondar	Version 1.0.6.x Revision E; Updated Trademarks page.
13-Apr-2015	WZhang	Version 1.0.6.x Revision F; Update terminology
01-Jun-2015	JMuraski	Version 1.1.0.x Updated with skeleton version 3.0.42
11-Aug-2015	TWilliams	Version 1.1.0.x. Added text to more clearly define how to configure a service using Local System

PI System Explorer

User Guide

For PI Asset Framework 2.10 included with PI Server 2018

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Fax: (01) 510-357-8136
Web: http://www.osisoft.com

PI System Explorer User Guide

© 2009-2018 by OSIsoft, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of OSIsoft, LLC.

OSIsoft, the OSIsoft logo and logotype, Managed PI, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, RLINK, and RtReports are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Version: 2.10

Published: 08 May 2018

Contents

Introduction to PI Asset Framework 1
How assets are represented in PI AF 1
PI System Explorer 2
PI System Explorer user interface components 3
Feature suggestions 4
PI System Explorer browser and navigator 4
Customize the navigator 5
PI System Explorer customization options 5
Column display configuration 7
Select multiple objects in the viewer 9
Object duplication 9
Icons that indicate object states 10
Check-in of database changes 11
Keyboard shortcuts 12
Change the language 13
Valid characters in PI AF object names 14
PI time 14

Server and database connections 19
PI AF server connections 19
Connect to a PI AF server 20
Add a PI AF server to the connection list 21
PI AF server properties 22
View general properties of the connected PI AF server 22
Manage identities 23
Manage mappings 24
View PI AF server object counts 26
PI Data Archive connections 26
Connect to PI Data Archive 26
Add PI Data Archive servers to connected server lists 28
View buffering status for connected PI Data Archive servers 29
PI AF database connections 30
Create PI AF databases 30
View or edit properties of PI AF databases 31
Connect to a database on a different PI AF server 32
Set the default database 33
Rename databases 33
Search for PI AF databases 33
Refresh the list of databases 34
Database deletion 34
PI AF and PI Data Archive collective connections 34
Manage connection preferences for PI System Explorer 34
Switch collective members 37
Switch to a specific collective member 37

Database import and export 39
Export a database to XML 39
XML export options 40

All Referenced Objects.... 40
File format for export and restore.... 43
Restore databases from exported XML files.... 44
Export of library objects to another database.... 45
PI AF saved libraries.... 45
Save databases as libraries.... 46
Load database libraries.... 46
Review properties of loaded libraries.... 46
AFExport utility.... 47
Guidelines for exporting collections.... 47
AFExport utility parameters.... 47
Sample export paths.... 49
AFImport utility.... 50
AFImport utility parameters.... 50

Retrieval of asset information.... 53
Asset browsing.... 53
Element browsing.... 54
Configure browser page size.... 54
Open additional browser windows.... 55
Asset and asset data searches.... 55
Quick search.... 56
Search result paging.... 56
Maximum query sizes.... 57
Searches on a specified date.... 57
Search for PI points.... 57
Manage search results.... 63
Search for elements.... 64
Special characters in name searches.... 69
Search for attributes on elements.... 69
Search for event frames.... 71
Search for attributes on event frames.... 74
Search for transfers.... 76
Search for attributes on transfers.... 77
Trends in PI System Explorer.... 79
Create trends.... 79
Version management for equipment and processes.... 80
Create versions for elements or models.... 80
Create table versions.... 81
Compare two versions.... 81
Display of different object versions and obsolete objects.... 82
Query Date and PI System Explorer time.... 83
Show specific dates and times.... 83
Time context for attribute values.... 84
View time series data.... 85
Boundary type.... 86
Filter expression.... 86
Statistics.... 87
Weighting.... 87
Event horizon mode.... 88
Restrictions on viewing time series data.... 89

Organization of asset models in PI AF.... 93

The structure of PI AF asset models....93
The design of PI AF asset hierarchies....94
Element references in the asset hierarchy....95
Reference types....96
Create single element references....98
Create multiple element references....98
Locate other references to the same element....99
Change reference types....99
Creation of custom reference types....100
Categorization of objects....100
Create new categories....101
Assign objects to categories....102

Representation of assets in PI AF....103
Asset representation with templates....103
Template strategy....104
Element templates....104
Default attribute....105
Base and derived templates....105
Extensions....106
Naming patterns....107
Create element templates....109
Find template references....111
Define templates for other objects....113
Child templates....113
Create child template references....114
Asset representation with elements....114
Create elements based on a template....115
Create elements not based on a template....116
Find where an element is used....116
Deletion of elements....117
Attribute-value reset to original properties....117
Convert elements to element templates....118
Change element templates....118
Storage of application-specific information....119
Create extended properties....119

Association of data with PI AF assets....121
Attribute templates....121
Attribute properties....122
Attribute traits....124
Control the display of attribute and attribute template values....134
Create attribute templates....135
PI AF data types....137
Define constant values for attribute templates....138
Add attribute extensions to template-based elements....140
Enumeration sets....142
Manage enumeration sets....144
Renumber enumeration values....146

Configuration of data references....149
Attribute configuration strings in PI System Explorer....149
Configuration strings for PI point data references....150

Path syntax for references to other attributes... 153
Substitution parameters in data references... 156
Symbols used in substitution parameters... 157
Name substitutions... 157
References to attribute values... 158
List of PI AF substitution parameters... 158
Format strings for time substitution parameters... 162
Find default PI Data Archive servers... 163
PI point data references... 164
Direct PI point references... 165
Indirect PI point references... 165
Create direct PI point data references from Tag Search results... 166
Configure direct or indirect PI point data references... 167
Unit of measure considerations... 168
Configuration of retrieval methods for attribute values... 168
PI point value updates from a data reference... 176
Attribute-value updates from PI point data references... 177
Configure automatic creation of PI points... 178
Edit PI point properties... 180
Preview substitution parameters in PI point data references... 180
PI point array data references... 181
Create PI point array data references... 181
Formula data references... 182
Formula data reference operators... 182
Formula data reference functions... 183
Units of measure in formula data references... 185
Configure formula data references... 185
String Builder data references... 188
Attribute references in String Builder... 188
Function implementation in String Builder... 189
Create String Builder data references... 192
Table lookup data references... 195
Create PI AF tables... 196
Create table column definitions... 198
Populate tables manually... 199
Data references from outside the PI System... 200
Configure table lookup data references... 211
URI Builder data references... 219
Create URI Builder data references... 220

Units of measure in PI AF... 225
Case sensitivity of UOM abbreviations... 226
Configuration of case-sensitive UOM abbreviations... 227
Base classes and derived classes... 228
UOM base classes... 228
Common UOM derived classes... 228
Create UOM classes... 229
Create units of measure... 230
UOM conversion calculation with a formula... 232
Calculate conversion values for a UOM... 233
UOM groups... 233
Manage UOM groups... 234
Origin of UOMs... 235

Security configuration in PI AF.....237
PI AF identities and mappings.....237
Built-in PI AF identities.....239
Security hierarchy.....240
Security inheritance of PI AF objects.....241
Permission inheritance of element objects.....242
PI AF access rights.....246
Deny permission option.....248
Security Configuration window.....249
PI AF server security.....250
Configure security for a PI AF server.....251
PI AF database security.....252
Configure security for new PI AF databases.....252
Configure security for a single PI AF database.....253
PI AF collection security.....254
Configure security for collections.....254
PI AF object security.....256
Configure security for objects.....256
Configure security for analyses and analysis templates.....257
Configure security for the UOM database.....258
Differences from PI Data Archive security model.....258

Event frames in PI AF.....261
Structure of event frames.....262
Advantages of event frames.....262
Examples of event frames.....263
Event Frames or Batch?.....264
Ways to create event frames.....264
Visualization of event frames.....265
Event frame templates.....266
Primary referenced element.....267
Data references to attributes in the primary referenced element.....267
Data references to attributes in other elements.....267
Create event frame templates.....269
Change event frame templates.....272
Create event frames.....273
Create attributes on event frames.....275
Value capture for event frames and transfers.....276
Capture values.....276
Lock event frames or transfers.....277
Acknowledgement of event frames.....278
Acknowledge event frames.....279
Transfers.....279
Create transfer templates.....280
Create transfers.....282
Create transfer ports.....283

Annotations in PI AF.....285
Element annotations.....285
Add annotations to elements.....286
Event frame annotations.....287
Add annotations to event frames.....287

Transfer annotations ... 288
Add annotations to transfers ... 288

Asset analytics ... 291
Introduction to analyses ... 291
Configure and manage analyses on an element ... 292
Analysis templates ... 294
Updates of analyses and analysis templates ... 295
Excluded attributes in analyses ... 295
Analysis scheduling ... 296
Expressions for analyses ... 298
Future data and analyses ... 301
Expression analyses ... 302
Create an expression analysis template ... 302
Create an expression analysis ... 305
Rollup analyses ... 307
Rollup analysis functions ... 308
Create a rollup analysis template ... 308
Create a rollup analysis ... 312
Event frame generation analyses ... 315
Implicit modes of event frame generation in asset analytics ... 316
Create an event frame generation analysis template ... 317
Create an event frame generation analysis ... 320
Start and end trigger conditions ... 322
Child event frame generation with multiple start triggers ... 323
SQC analyses ... 324
Create an SQC analysis template ... 325
Create an SQC analysis ... 327
Sample analyses ... 329
Build an expression analysis to study efficiency deviation ... 330
Create a template for power rollup analyses ... 332
Create event frames automatically to track inefficiency ... 334
Backfilling and recalculation of analyses ... 336
Backfill or recalculate data for an analysis ... 337
Backfill or recalculate data for multiple analyses ... 338
Reconciliation of event frames during backfilling ... 339
Configure automatic recalculation of analyses and analyses templates ... 340
Enable or disable automatic recalculation for multiple analyses ... 341
Open recalculation log folder ... 341
Management of analyses in a PI AF database ... 341
Expression functions reference ... 342
Abs ... 349
Acos ... 350
AND ... 350
ArrayLength ... 351
Ascii ... 351
Asin ... 352
Atn ... 353
Atn2 ... 353
Avg ... 354
BadVal ... 355
Bod ... 356
Bom ... 356

Bonm..... 357
Ceiling..... 358
Char..... 359
Compare..... 359
Concat..... 360
Contains..... 361
Convert..... 362
Cos..... 362
Cosh..... 363
Cot..... 364
Coth..... 364
Cov..... 365
Csc..... 366
Csch..... 367
Curve..... 367
Day..... 368
DaySec..... 369
DeltaValue..... 369
DigState..... 370
DigText..... 371
E..... 371
ELSE..... 372
EventCount..... 372
EventFrame..... 373
Exp..... 374
FilterData..... 375
FindEq..... 375
FindGE..... 376
FindGT..... 377
FindLE..... 378
FindLT..... 379
FindNE..... 380
FirstValue..... 381
Float..... 381
Floor..... 382
Format..... 383
Frac..... 383
HasChanged..... 384
HasValueChanged..... 385
Hour..... 385
IF..... 386
IN..... 387
InStr..... 387
Int..... 388
InterpolatedValues..... 389
IsSet..... 390
LastValue..... 391
LCase..... 392
Left..... 392
Len..... 393
LinRegr..... 393
Log..... 395
Log10..... 395

Logbase....396
LTrim....397
MapData....397
Max....398
Median....399
Mid....400
Min....401
Minute....402
Mod....402
Month....403
NextEvent....403
NextVal....404
Noon....405
NoOutput....405
NormalIrnd....406
NOT....406
NumOfChanges....407
OR....407
ParseTime....408
PctGood....409
Pi....409
Poisson....410
Poly....410
Pow....411
PrevEvent....412
PrevVal....412
PStDev....413
Rand....414
Range....415
Rate....416
RecordedValues....416
RecordedValuesByCount....417
Remainder....419
Right....420
Round....420
Roundfrac....422
RTrim....423
Sec....423
Sech....424
Second....425
SecSinceChange....425
Sgn....426
Sin....426
Sinh....427
Sqr....427
StateNo....428
SStDev....429
StDev....430
String....431
TagAvg....432
TagBad....433
TagDesc....433
TagEU....434

TagExDesc ... 435
TagMax ... 435
TagMean ... 436
TagMin ... 437
TagName ... 438
TagNum ... 439
TagSource ... 439
TagSpan ... 440
TagTot ... 440
TagType ... 442
TagTypVal ... 443
TagVal ... 443
TagZero ... 444
Tan ... 445
Tanh ... 445
Text ... 446
THEN ... 446
TimeEq ... 447
TimeGE ... 448
TimeGT ... 449
TimeLE ... 450
TimeLT ... 451
TimeNE ... 452
TimeStamp ... 452
Total ... 453
Trim ... 454
Trunc ... 455
UCase ... 456
Weekday ... 456
Year ... 457
Yearday ... 457
Steam functions for analysis expressions ... 458
Steam functions in asset and tag-based analytics ... 459
Ranges for steam function inputs ... 460
Unit of measure conversion for steam functions ... 462
Steam property reference states ... 463
Steam functions reference ... 463
Asset analytics native integration of external systems ... 477
Integration of MATLAB Production Server into asset analytics ... 478
Configure an external system endpoint ... 479
Supported functions and data types in MATLAB Production Server ... 479
Error handling ... 480
PI Analysis Service management ... 481
View analysis service statistics ... 481
View or modify analysis service ... 482
Analysis service configuration ... 482
Buffering PI point outputs for PI Analysis Service ... 486

Notifications ... 487
Introduction to notifications ... 488
Notifications and event frames ... 488
Notifications history ... 489
Requirements for notifications ... 490

Client support for notifications 490
Notifications security..... 490
Security for contacts..... 491
Security for notification rules and templates..... 492
Configure authentication for SMTP server connection..... 493
Configure authentication for a web service connection..... 494
Create a notification rule..... 495
Create a notification rule template..... 495
Trigger criteria in notification rules..... 496
Non-Repetition Interval in notification rules..... 498
Resend Interval in notification rules..... 498
Delivery formats in notification rules..... 498
Add a domain to the whitelist..... 500
Add a screenshot in notification email delivery format..... 502
Subscriptions in notification rules..... 503
Customization of subscription content and delivery..... 504
Configuration of notifications delivery endpoints..... 504
Configure an email delivery endpoint..... 505
Configure a dynamic email delivery endpoint..... 506
Configure a REST web service delivery endpoint..... 506
Configure a SOAP web service delivery endpoint..... 508
Contacts..... 509
Configure Active Directory access for contacts..... 511
Manage contacts..... 512
Options for the notifications email delivery channel..... 516
Options for the notifications web service delivery channel..... 516
Configuration of notification rules for analyses or event frames..... 517
Configure notification rules from analyses..... 518
Configure notification rules for user-defined event frames..... 521
Configure escalations for notification rules..... 524
Management of notification rules..... 524
Changes from PI Notifications 2012..... 524
Migration from PI Notifications 2012..... 525
Additional resources..... 528

Management of analyses and notification rules..... 529
Search for analyses or notification rules..... 529
Management of analyses..... 530
Management of notification rules..... 532

Process models in PI AF..... 533
The scope of a PI AF model..... 533
Guidelines for PI AF models..... 534
Submodels..... 534
Element types in models..... 535
Create PI AF models..... 535
Edit PI AF models..... 536
Ports and connections..... 536
Create ports..... 536
Create connections..... 537
Layers..... 537
Create layers..... 538

PI AF utilities and plug-ins **539**
Launch PSE with command line options 539
AFExplorer parameters 539
PI AF Diagnostics utility 540
Grant permissions for PI AF Diagnostics utility 541
Run the AFDiag utility 542
AFDiag utility parameters 543
Audit Trail implementation 548
AF Update Plug-in Configurations utility 549
Set PI AF server utility 551
SetPISystem utility parameters 551
Capture AF SDK event trace output 552
AFGetTrace utility parameters 552
Track PI AF changes with Audit Trail 553
Review changes with the Audit Trail utility 554
View installed plug-ins 556
Command-line plug-in registration 556
RegPlugIn parameters 558
Create an XML registration file 559
Create an SQL registration script 560
Register plug-ins with generated SQL scripts 561
Plug-in provider management 561
Locate the names of trusted providers 562

Technical support and other resources **563**

Introduction to PI Asset Framework

With PI Asset Framework you can represent assets and processes as PI AF objects and structure them to provide value to your business.

PI System Explorer and PI Builder are the primary tools that you use to create and manage PI AF objects. Both tools include support for the following features:

- **Event frames**

  With event frames, you can capture important process events and collect relevant data around them to help analyze why they occurred. See Event frames in PI AF for more information.

- **Asset analytics**

  Integrated into PI System Explorer, you can use asset analytics to create calculations and set up conditional statements involving asset values. See Asset analytics for more information.

- **Notifications**

  Integrated into PI System Explorer in PI AF 2016 R2, you can use notifications to create rules by which users can be alerted in real time to anomalous conditions in the system. See Introduction to notifications for more information.

Topics in this section

- How assets are represented in PI AF
- PI System Explorer

How assets are represented in PI AF

PI Asset Framework (PI AF) enables you to build a representation of your equipment and processes that can give you tremendous insight into your data. In PI AF, the equipment and processes that you want to monitor are called assets. The PI AF representation of all your assets and processes together is called an asset model. The asset model organizes all your equipment into a structure that makes it easy to find information.

In the PI AF asset model, each piece of equipment is represented by an element. The associated data for an element is stored as attributes on the element. Attributes can hold simple values, representing fixed information, such as the diameter of a tank. An attribute can alternatively reference a PI point, a formula, a value from a relational database, a file, a photograph, and more. All relevant data about an asset is tied to the element representing that asset.

For example, suppose you have a pump with three associated pieces of data: the pressure (read from a PI point), the inlet temperature, and the outlet temperature. To model this in PI AF, you can create a PI AF element to represent the pump and then create three attributes to represent the associated data.

The following illustration shows how the data might look in PI AF. Although all the values are PI point values, the user never needs to know the names of the PI points. All the data is available directly on the element.

Name	Value
Head Pressure	92.1994592
Temperature Inlet	15 deg C
Temperature Outlet	16 deg C

Video

For information on assets and attributes, watch this video:

https://www.youtube.com/watch?v=18RgS8Y3JtQ&rel=0

PI System Explorer

PI System Explorer provides a graphical user interface for creating, editing, and managing PI AF objects. Use PI System Explorer to create and manage your asset framework, including PI AF databases, elements, and templates.

PI System Explorer versions

On computers with a 64-bit operating system, both a 32-bit and a 64-bit version of PI System Explorer are available after a PI AF client installation. Beginning with PI AF 2017, the 32-bit version is labeled **PI System Explorer Legacy 32-bit**, whereas the 64-bit version is labeled simply **PI System Explorer**.

The legacy 32-bit version is available should you need to continue to use the legacy version of Notifications (prior to the 2016 R2 version) and its contact editor. This version of the contact editor is required to work with legacy delivery channels. For more information on notifications, see Notifications.

Support for FIPS

Beginning with PI AF 2017 R2, PI AF supports Windows environments that enforce the use of U.S. Federal Information Processing Standards (FIPS) cryptographic algorithms. No additional configuration is required once the local security policy **System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing** has been enabled.

To be in compliance with FIPS, users who need to use passwords to link to external tables can only use PI AF 2017 R2 or later clients and servers. Furthermore, users cannot export data from a PI AF database that contains table passwords created with a PI AF 2017 R2 or later client and import those passwords to PI AF clients on older releases. For more information on linking to external table data references, see Data references from outside the PI System.

Topics in this section

- PI System Explorer user interface components
- Feature suggestions
- PI System Explorer browser and navigator
- Customize the navigator
- PI System Explorer customization options

- Column display configuration
- Select multiple objects in the viewer
- Object duplication
- Icons that indicate object states
- Check-in of database changes
- Keyboard shortcuts
- Change the language
- Valid characters in PI AF object names
- PI time

PI System Explorer user interface components

The following illustration shows the major components of the PI System Explorer.

1. Menu bar
2. Toolbar
3. Browser
4. Navigator
5. Status bar
6. Palette
7. Viewer

Videos

For information on PI System Explorer browser and navigator, watch this video:

https://www.youtube.com/watch?v=Dtf7xC2N7ak&rel=0

For information on PI System Explorer palette and viewer, watch this video:

https://www.youtube.com/watch?v=IXXR88qBv2I&rel=0

Feature suggestions

Beginning with PI AF 2017, you can make suggestions on features you would like to see added to the PI System. You can also review suggestions that other users have already submitted and vote for those you approve of. You can select from the following categories for your suggestion:

- Analytics and Calculations
- Asset Framework
- Data Archive
- Event Frames
- General
- Help/Documentation/Videos
- Installation
- Notifications
- PI Builder
- Security
- System Management

To provide feedback, you select **Help** > **Give Us Feedback** and click **Sign in**. You can type your feedback into the **Enter your idea** field, select an appropriate category, and provide additional details as needed.

PI System Customer Experience Improvement Program

The **Customer Experience Feedback Option** on the **Help** menu enables you to participate in the telemetry program that OSIsoft uses to collect anonymous usage data to improve the PI System and to prioritize new features. The collected data does not include business data or logic, but can include information such as IP addresses, host names, and names of PI System objects.

To participate in the telemetry program, you select **Yes, I would like to participate (Recommended)** and click **OK**.

PI System Explorer browser and navigator

The content of the PI System Explorer browser depends on which button is selected in the navigator. The default available objects in the navigator are elements, event frames, library objects, units of measure, and contacts. The management object is also displayed if the Management plug-in (which encompasses analyses and notification rules) is installed.

> **Note:**
> If legacy notifications are installed on a PI AF client computer, MyPI and Notifications may also be displayed in the navigator.

You can move very quickly between browser objects in PI System Explorer with the following keyboard shortcuts. Notice also that different colors are displayed at the top of the browser and viewer to help distinguish between selected browser objects:

Shortcut	Browser
CTRL+1	Elements
CTRL+2	Event Frames
CTRL+3	Library
CTRL+4	Unit of Measure
CTRL+7	Contacts
CTRL+0	Management

Customize the navigator

To increase the space available on your screen for objects in the browser, change the buttons in the navigator to icons.

Procedure

1. Right-click in the navigator and click **Navigation Pane Options**.
2. In the Navigation Pane Options window, clear the check box beside each component on the **Display Full Button** list.
3. Click **OK**. The rows of button labels are replaced by a single row of icons.

4. Move your cursor over an icon to display its label.

PI System Explorer customization options

You can customize many aspects of PI System Explorer to suit your needs with **Tools** > **Options**.

General tab

You use the **General** tab to control display options for several features. You can control:

- Keystroke to open Check-In and Undo-Checkout windows. For more information, see Check-in of database changes.
- Title bar appearance.
- Page size for browser objects. For more information, see Configure browser page size.
- Number of queries returned in object searches. For more information, see Search result paging.

- Unit of measure appearance for attributes. For more information, see Show attribute values in source unit of measure.
- Display of attribute values to the units of measure that are mapped to a selected UOM group. When no UOM group is selected, attribute values are based on the default UOM defined in attribute templates or their source unit of measure (if **Use Source Unit-Of-Measure for attribute display** is selected). For more information, see UOM groups.
- Display of excluded attributes. For more information, see Excluded attribute property.
- Number of digits displayed for attribute values. For more information, see Control the display of attribute and attribute template values.

Time Context tab

You use the **Time Context** tab to define the time or time range that PI System Explorer uses to display attribute values. For more information, see Set time context for displayed attribute values.

Server Options tab

You use the **Server Options** tab to define how PI System Explorer should connect to a PI AF collective or PI Data Archive collective. For more information, see Manage connection preferences for PI System Explorer.

Topics in this section

- Show attribute values in source unit of measure
- Change the unit of measure for displayed attribute values

Show attribute values in source unit of measure

For attributes that are defined by data references, you can have their values displayed in the units of measure that are defined by the data reference, rather than in the default units of measure that are established in attribute templates.

Procedure

1. Select **Tools** > **Options**.
2. On the **General** tab, select **Use Source Unit-Of-Measure for attribute display**.
3. Click **Apply**.
4. Click **OK** to close the window.

Change the unit of measure for displayed attribute values

During your current session in PI System Explorer, you can change the unit of measure (UOM) that is displayed for an attribute value from its default or mapped value.

Procedure

1. In the **Attributes** tab of the **Elements** or **Event Frames** viewer, right-click the value for an attribute and click **Change Display UOM**.
2. In the Select Display UOM window, select a new UOM for the value from the **Display UOMs** list.

   Note:
   The list contains all UOM values in a class. **<Default>** is displayed beside the default UOM. If you have selected a UOM group in **Tools** > **Options** and the selected UOM group has a mapping for the attribute's default UOM, **<Mapped>** is displayed beside the mapped UOM.

3. Click **OK**.

Column display configuration

Property columns

Whenever data columns are displayed in the viewer or a search results window, you can customize which property columns you wish to view. At the far right of the window, you can click and select additional column selections or clear default column selections.

Attribute columns

In viewer or search results windows for elements and event frames, you can also select attribute columns and display them as columns alongside the default property columns. An attribute column is identified by the **|** character.

Select attributes to display as viewer columns

For collections of **Elements**, **Element Searches**, **Event Frames Searches**, and **Transfer Searches**, you can select attributes to display as columns in the viewer, in addition to default properties.

Procedure

1. In the navigator pane, choose one of the following actions:
   - To select attributes for **Elements** and **Element Searches** collections, click **Elements**.
   - To select attributes for **Event Frames Searches** and **Transfer Searches** collections, click **Event Frames**.
2. In the browser, click the collection for which you want to select attributes.
3. At the far right of the viewer, click .
4. Click **Select Attributes**.
5. In the Select Attributes window, choose from the following options to select the attributes that you want displayed in the viewer:

To ...	Do this ...
Add attributes from a template	a. Select the **Add Attributes from Template** option. b. Select a template from the list.
Add attributes from an element or event frame **Note:** You cannot add attributes from a **Transfer Searches** collection	a. Select the **Add Attributes from Element/ Event Frame** option. b. Choose from the following actions: ▪ To search for an element or event frame, click and enter criteria to locate attributes in the Element Search or Event Frame Search window. ▪ To select an element or event frame from the browser tree, click and select from the Element Browser or EventFrame Browser window. c. Click **OK**.
Add other attributes	a. In the **Others** field, enter attribute names separated by a semicolon, or click and enter criteria to locate attributes in the Attribute Search window. b. Click **Add**.

Selected attributes are displayed in alphabetical order in the left-hand **Attributes** or **Attribute Templates** list. For added convenience, you can group them by category or template by selecting one of the **Group by** check boxes. You can also filter them if necessary with further search criteria.

6. From the **Attributes** or **Attribute Templates** list, select the attributes you want to display in the viewer.
   a. Use standard Windows selection keystrokes (such as SHIFT+<click> and CTRL+<click>) to select contiguous and non-contiguous attributes in the list, as described in Select multiple objects in the viewer.
   b. Click to move selected items or to move all items to the right-hand **Attributes** list.
7. Optional. Adjust the order that attributes will be displayed in viewer columns:

To ...	Do this ...
Move an attribute up the list	Select an attribute below the top row and click .
Move an attribute down the list	Select an attribute above the bottom row and click
Remove an attribute from the list	Select an attribute and click .
Remove all selected attributes	Click .

8. Optional. You can add additional attributes as required, for example from another template. Repeat steps 5 to 7.
9. Click **OK** to complete the attribute selection.

The selected attributes are displayed as columns in the viewer.

10. Optional. You can control which selected attribute columns are displayed in the viewer:

To ...	Do this ...
Remove an attribute column	a. Click ⚙. b. Click **Attribute Columns**. c. Clear the attribute you want to remove.
Remove all attribute columns	a. Click ⚙. b. Clear **Show Attribute Columns**.

Select multiple objects in the viewer

In the viewer, use standard Windows mouse and keyboard combinations to select more than one object to complete operations such as copying, deleting, exporting, checking items in and out, as well as changing templates.

Procedure

- To select a group of contiguous objects in the viewer, click one object, move the cursor to the end of the group, press SHIFT and click the last object in the group.
- To select non-contiguous objects, press the CTRL key as you click each object in the viewer.
- To select all objects in the viewer, press CTRL+A. To deselect individual objects in the selection, press CTRL and click each object.

Results

After selecting the objects, right-click and click the operation to perform on the context menu.

Object duplication

You can copy individual objects or multiple rows of objects and paste the information for those objects into a spreadsheet in a tab-separated format.

The **Copy** option is available on context menus throughout PI AF. The **Copy Path** and **Copy Cell** options are available where appropriate.

> **Note:**
> When you drag and drop, the clipboard always contains the path information, which renders the data compatible with other client applications, such as PI ProcessBook.

Copy

In any PI AF browser, use **Copy** on a context menu to place the full path of the selected object on the clipboard:

```
\\Kaboom\Chocolate Milk Tutorial\ChocolateMilkModel\MixFlow
```

In any viewer grid or list, use **Copy** to place the content of each displayed column in the selected row on the clipboard. If you select multiple PI AF rows, **Copy** places multiple data

rows separated by a new line. In attributes lists, the copied data includes strings that correspond to the Configuration (), Quality (), Template (), and Hidden () column icons, in addition to other displayed columns:

```
1          2  3    4        5  6        7
Is Not Configuration Item Good Tank Is Not Hidden Level 0 ft MilkTank2|Level
```

1	
2	
3	
4	
5	**Name**
6	**Value**
7	**Path**

Copy Path

The **Copy Path** option places the full path for each selected object on the clipboard. If multiple PI AF objects are selected, **Copy Path** places multiple paths separated by a new line:

```
\\ABACUS-CURRENT\NuGreen\NuGreen\Houston\Cracking Process\Equipment\F-321|Model
\\ABACUS-CURRENT\NuGreen\NuGreen\Houston\Cracking Process\Equipment\F-321|Motor
Amps
```

Copy Cell

In any viewer grid or list containing multiple columns, such as attributes, referenced elements, child event frames, and so on, the **Copy Cell** option places the contents of the selected cell on the clipboard.

Icons that indicate object states

PI System Explorer uses the following (mini) icons superimposed over object icons, such as attributes (), elements (), event frames (), and servers (), to indicate the current state of those objects. Such icons can also be combined to indicate multiple states: for example, uses the referenced and default icons to indicate the primary referenced element in an event frame, whereas uses the connected and default icons to indicate the default connected PI AF server.

Icon	Object state
	Is changed, but not yet applied.
	Is checked out by you.
	Is checked out by a different user.
	Is available in multiple versions.
	Is a reference to another object.
	Is the default.

Icon	Object state
	Is disconnected, such as a collective member.
	Is connected, such as a PI AF server.
	Is a search.
	Is a template.
	Is arranged by category.
	Is unsaved, such as a search.
	Is unsupported, such as a server version.

Check-in of database changes

Whenever you make a change to an object, including whenever you create a new object, PI System Explorer checks that object out from the database. You need to save these changes in the database, although you can apply a change to a selected object by clicking ✓ in the toolbar and then check in your changes later.

- Unsaved changes are indicated by a red check mark icon.
- Changes that have been applied, but not saved to the database are indicated by a dark red check mark icon.

As you work, you can choose from these options to save your changes:

- Click **File** > **Check In**.
- On the toolbar, click **Check In**.
- In the browser, right-click an object and click **Check In** to save changes in the database for that object only.

  The first two options allow you to save changes for all modified objects. The Check In window opens and displays objects that have been modified. You can select them all in or select some to check in, and allow others to remain checked out. Click **Session** to select objects modified only in the current session. You can still check in objects modified in other sessions.

Undo Check Out

If you decide not to save your changes, you can use the **File** > **Undo Check Out** option. In the Undo Check Out window, you clear any items in the list you *do* want to check in, and click **Undo Check Out**.

Undo Check Out for all users

Occasionally, you may be unable to modify a PI AF object because the element is checked out to another user. You see the following message types that inform you that an object is locked by a user:

- In notifications, when you attempt to stop, rename, delete, or otherwise change a notification, the error message `Notification 'Notification_name' with Unique ID 'ID' is locked by user Windows_account` is displayed.
- In PI AF, when you attempt to work with an object, the error message `AF Object Name 'Object_name' with UniqueID 'ID' is locked by user Windows_account` is displayed.

To undo the lock, a user who has the Admin permission on the object set to Allow can select **Show All Users** in the Undo Check Out window, select **All** and then click **Undo Check Out**. For more information, see PI AF access rights.

Videos

For information on the check in/check out mechanism of a PI AF object, watch this video:

https://www.youtube.com/watch?v=Pb8ZKxOaTsE&rel=0

For information on how to check in modifications to PI AF objects, watch this video:

https://www.youtube.com/watch?v=Ky1ICZwwBqY&rel=0

For information on how to undo check out of PI AF objects, watch this video:

https://www.youtube.com/watch?v=OPSy_HbvAyY&rel=0

Keyboard shortcuts

PI System Explorer provides keyboard shortcuts for navigation and other tasks that require a mouse or other pointing device.

Shortcut	Action
CTRL+A	Selects all objects in the viewer.
CTRL+C	Copies the selected object to the clipboard.
CTRL+ALT+C	Copies the path of the selected object to the clipboard.
CTRL+V	Pastes the object on the clipboard to the viewer.
CTRL+X	Cuts (deletes) the selected object and copies it to the clipboard.
DELETE	Deletes the selected object.
SHIFT+DELETE	Same as CTRL+X
INSERT	Adds a new object to the viewer or browser.

Shortcut	Action
HOME	Selects the first row in the viewer, for example, the first row in a table of attributes.
END	Selects the last row in the viewer.
CTRL+HOME	Selects the first cell of the current page in the viewer.
CTRL+END	Selects the last cell of the current page in the viewer.
ALT+HOME	Selects the first page of objects in the viewer.
ALT+END	Selects the last page of objects in the viewer.
CTRL+PAGE UP	Selects the previous page of objects in the viewer.
CTRL+PAGE DOWN	Selects the next page of objects in the viewer.
CTRL+ENTER	If the viewer contains multiple pages of objects, displays the Select Page Number window.
ALT+ENTER	In the browser, displays the properties of the selected object.
SPACE or ENTER	Presses the currently selected button.
Left, Right, Up, and Down Arrows	Navigate objects in the viewer or browser.
F2	Edits the selected object on the viewer. For complex objects, displays the edit window for the object.
F4 or ALT+Up Arrow or ALT+Down Arrow	Displays the choices in the selected list box. For layered lists, displays the complete hierarchy of choices.
CTRL+1	Navigates to the Elements browser.
CTRL+2	Navigates to the Event Frames browser.
CTRL+3	Navigates to the Library browser.
CTRL+4	Navigates to the Unit of Measure browser.
CTRL+5	Navigates to the MyPI browser.
CTRL+6	Navigates to the Notifications browser.
CTRL+7	Navigates to the Contacts browser.
CTRL+0	Navigates to the Analyses browser.

Change the language

You can change the language for PI System Explorer on your computer if you have a Language Pack and the desired language resources have been installed. The language setting is per user

locale, so if others want to use PI System Explorer on the same computer under a different login, they can use different language resources if available.

Procedure

1. Run the Language Pack and select the language resources you want to install, if they are not already available.
2. Click **View** > **Language Settings**.
3. In the Language Settings Tool window, select the target language and click **OK**.
4. Quit and restart PI System Explorer.

   PI System Explorer appears in the specified language.

   Note:
   The language setting is account-specific, so users who log in under a different account see the language specified for that account.

Valid characters in PI AF object names

PI AF has the following constraints on object names:

- Leading or trailing spaces are removed from names.
- Maximum name length is 259 characters.
- Blank names are not permitted.

You can use standard keyboard characters, with the following exceptions:

Object type	Invalid characters
Other than contacts	Control characters plus: * ? ; { } [ ] | \ ` ' "
Contacts	Control characters plus: * ? ; [ ] | \ "

PI time

You can use a special syntax, called PI time, to specify inputs for time stamps and time intervals. PI time uses specific abbreviations, which you combine to create time expressions.

Topics in this section

- PI time abbreviations
- PI time expressions
- Time-stamp specification
- Time-interval specification

PI time abbreviations

When specifying PI time, you can use specific abbreviations that represent time units and reference times.

Time-unit abbreviations

Abbreviation	Full version	Plural version	Corresponding time unit
`s`	`second`	`seconds`	Second
`m`	`minute`	`minutes`	Minute
`h`	`hour`	`hours`	Hour
`d`	`day`	`days`	Day
`mo`	`month`	`months`	Month
`y`	`year`	`years`	Year
`w`	`week`	`weeks`	Week

To specify time units, you can specify the abbreviation, the full version, or the plural version of the time unit, such as `s`, `second`, or `seconds`. You must include a valid value with any time unit. If specifying seconds, minutes, or hours, you can specify a fractional value, such as `1.25h`. You cannot specify fractional values for other time units.

Reference-time abbreviations

Abbreviation	Full version	Corresponding reference time
`*`		Current time
`t`	`today`	00:00:00 (midnight) of the current day
`y`	`yesterday`	00:00:00 (midnight) of the previous day
`sun`[1]	`sunday`	00:00:00 (midnight) on the most recent Sunday
jun[2]	`june`	00:00:00 (midnight) on the current day in June of the current year
`dec` *DD*	`december` *DD*	00:00:00 (midnight) on the *DD*th day of December in the current year
YYYY		00:00:00 (midnight) on the current day and month in year *YYYY*
M-D or *M/D*		00:00:00 (midnight) on the *D*th day of month *M* in the current year
DD		00:00:00 (midnight) on the *DD*th day of the current month

[1]: Use the first three letters as an abbreviation for any day of the week: `sun`, `mon`, `tue`, `wed`, `thu`, `fri`, or `sat`.

[2]: Use the first three letters as an abbreviation for any month of the year: `jan`, `feb`, `mar`, `apr`, `may`, `jun`, `jul`, `aug`, `sep`, `oct`, `nov`, or `dec`.

PI time expressions

PI time expressions can include fixed times, reference-time abbreviations, and time offsets. A time offset indicates the offset direction (either + or -) and the offset amount (a time-unit abbreviation with a value).

For example, PI time expressions can have the following structure:

Structure	Example
Fixed time only	`24-aug-2012 09:50:00`
Reference-time abbreviation only	`t`
Time offset only	`+3h`
Reference-time abbreviation with a time offset	`t+3h`

Include at most one time offset in an expression; including multiple time offsets can lead to unpredictable results.

Time-stamp specification

To specify inputs for time stamps, you can enter time expressions that contain:

- Fixed times

  A fixed time always represents the same time, regardless of the current time.

Input	Meaning
`23-aug-12 15:00:00`	3:00 p.m. on August 23, 2012
`25-sep-12`	00:00:00 (midnight) on September 25, 2012

- Reference-time abbreviations

  A reference-time abbreviation represents a time relative to the current time.

Input	Meaning
`*`	Current time (now)
`3-1` or `3/1`	00:00:00 (midnight) on March 1 of the current year
`2011`	00:00:00 (midnight) on the current month and day in the year 2011
`25`	00:00:00 (midnight) on the 25th of the current month
`t`	00:00:00 (midnight) on the current date (today)
`y`	00:00:00 (midnight) on the previous date (yesterday)
`tue`	00:00:00 (midnight) on the most recent Tuesday

- Reference-time abbreviations with a time offset

  When included with a reference-time abbreviation, a time offset adds or subtracts from the specified time.

Input	Meaning
`*-1h`	One hour ago
`t+8h`	08:00:00 (8:00 a.m.) today
`y-8h`	16:00:00 (4:00 p.m.) the day before yesterday
`mon+14.5h`	14:30:00 (2:30 p.m.) last Monday
`sat-1m`	23:59:00 (11:59 p.m.) last Friday

- Time offsets

Entered alone, time offsets specify a time relative to an implied reference time. The implied reference time might be the current clock time or another time, depending on where you enter the expression.

Input	Meaning
`-1d`	One day before the current time
`+6h`	Six hours after the current time

Time-interval specification

Time-interval inputs define intervals for collecting or calculating values during a time period. For example, you might specify a 60-minute interval to compute an hourly average over a 12-hour period. To specify time-interval inputs, enter a valid value and time unit:

- Positive values define intervals that begin at the earlier time in the period and that finish at or before the later time in the period.

Start time	`2:00:00`
End time	`3:15:00`
Time interval	`30m`
Returned intervals	2:00:00 to 2:30:00 2:30:00 to 3:00:00

- Negative values define intervals that finish at the later time in the period and that begin at or after the earlier time in the period.

Start time	`2:00:00`
End time	`3:15:00`
Time interval	`-30m`
Returned intervals	2:15:00 to 2:45:00 2:45:00 to 3:15:00

Server and database connections

After a PI AF Client installation, an administrator needs to configure a PI AF server for general use. Configuration includes the following tasks:

- Connect to a PI AF server and set up a default.
- Set up PI AF identities.
- Map Windows user accounts to one or more PI AF identities.
- Select a default PI Data Archive, if one was not selected during PI AF Client installation.
- Create (or import) PI AF databases.
- Specify PI System Explorer connection preferences, if PI AF collectives are defined.
- Configure access permissions for identities to the PI AF Client server and databases. For more information, see Security configuration in PI AF.

Video

For information on how to connect to and search a PI System, watch this video:

https://www.youtube.com/watch?v=_n3yLpjMhew&rel=0

Topics in this section

- PI AF server connections
- PI Data Archive connections
- PI AF database connections
- PI AF and PI Data Archive collective connections

PI AF server connections

An administrator needs to select the PI AF servers to which to connect and establish the default. Additionally, an administrator needs to set up PI AF identities and mappings to PI AF servers.

Topics in this section

- Connect to a PI AF server
- Add a PI AF server to the connection list
- PI AF server properties
- View general properties of the connected PI AF server
- Manage identities
- Manage mappings
- View PI AF server object counts

Connect to a PI AF server

Use the Servers window to review PI AF server connections and to connect to different servers as needed.

Procedure

1. From the PI System Explorer menu bar, choose **File** > **Connections**.

   In the Servers window, you see a list of known servers, identified by the following icons.

	A disconnected PI AF server in the Known Server Table.
	A connected PI AF server.
	The default connected PI AF server.
	A disconnected PI AF collective.
	A connected PI AF collective.
	A PI Data Archive in the Known Server Table.
	The default connected PI Data Archive.
	A disconnected PI Data Archive collective.
	A connected PI Data Archive collective.

> **Note:**
> The icon and a warning indicates that the PI AF client is connected to an unsupported version of PI Data Archive, such as version 3.4.375 or 3.4.370. The PI AF client cannot connect to a PI Data Archive server that is running software earlier than version 3.4.370.

2. To connect to a PI AF server, choose from the following actions:

To ...	Do this ...
Connect to a different PI AF server	Select a server in the list and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server in the list and click **Connect As**. b. In the Connect to PI AF Server *server* window, enter a Windows user account name and password. c. Click **OK**.
Connect to a server that is not displayed on the Known Servers Table	Click **Add Asset Server**. For more information, see Add a PI AF server to the connection list.
Set a default PI AF server	Right-click a connected PI AF server in the list and click **Set as Default AF Server**. A icon is displayed beside the server that is now the default.

Connect to a collective	a. Right-click a collective in the list. b. Choose one of the following actions: ▪ To connect to the primary server in the collective, click **Connect to Primary**. ▪ To connect to a specific collective member, click **Connect to Collective Member**, as described in Switch to a specific collective member.

3. Click **Close**.

Add a PI AF server to the connection list

If the PI AF server you want to connect to is not currently displayed on the Known Servers Table, you can add it in the PI AF Server Properties window.

Procedure

1. In PI System Explorer, click **File** > **Connections**.
2. Click **Add Asset Server**.
3. In the PI AF Server Properties window, enter the PI AF server properties.
4. The **Name** field does not have to match the host name. After you connect to a PI AF server, you can change the server name by clicking **Rename** and entering a new name.

   Caution:
   Renaming the PI AF server impacts all clients.

5. The **Account** field is typically not needed in normal installations and should be left blank.

   By default, PI AF clients such as PI System Explorer attempt to use a Service Principle Name (SPN) registered by the PI AF server to establish a secure connection to the server. If an SPN cannot be created by the PI AF server, a secure connection can be established via Microsoft NTLM if the server is running under a system account, such as Network Service. In each case, the **Account** field is not needed and should be left blank.

   If the PI AF server is run under a domain account, however, and an SPN is not successfully created by either the PI AF server or an administrator, the domain account of the server must be specified so that the PI AF client can securely identify the server.

6. The default **Timeout** value of 300 seconds is acceptable in most cases. If you experience timeout errors as you work in PI System Explorer, increase the time in the **Timeout** box.
7. Optional. **Aliases** are alternate names for the PI AF server which users can use to look for the server. When configured, PI AF server aliases are stored locally on the client only.
8. Optional. The **Configure Active Directory** link is for setting up the notifications contacts list. This is a PI AF system administrator function.
9. Click **Connect**.

   Note:
   If an error message opens saying that you cannot connect to the PI AF server, you need to enter the domain account of the server in the **Account** field.

10. After a server if connected, you can click **Security** and set up security for the server. For more information, see Security hierarchy.
11. Click **OK**.

PI AF server properties

You manage the configuration of the currently connected PI AF server in the PI AF Server Properties window. The following tabs are available:

- **General**. For more information, see View general properties of the connected PI AF server.
- **Plug-Ins**. For more information, see View installed plug-ins.
- **Libraries**. For more information, see Review properties of loaded libraries.
- **Identities**. For more information, see Manage identities.
- **Mappings**. For more information, see Manage mappings.
- **Counts**. For more information, see View PI AF server object counts.

View general properties of the connected PI AF server

On the **General** tab of the PI AF Server Properties window, review and modify settings for the currently connected PI AF server, such as:

- Server name, host name, and ID
- Server account
- Server version number
- Server port, timeout, and aliases

Procedure

1. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
2. On the **General** tab of the PI AF Server Properties window, review the server information and modify as needed.
3. You can also choose from the following actions.

To ...	Do this ...
Rename the PI AF server	a. Click **Rename**. b. In the **New Name** field of the New PI AF Server Name window, replace the current name with a new name. c. Click **OK**. **Caution:** Renaming the PI AF server impacts all PI AF clients.

Enter an alias	a. Click . b. In the **New Name** field of the New Alias Name window, enter an alternate name that users can use to look for the PI AF server. Aliases are stored locally on the client.
Configure Active Directory access for contacts	See Configure Active Directory access for contacts.
View or modify security permissions of mapped user identities	a. Click **Security**. b. In the Security Configuration window, review the permissions for the listed identities that have been configured for each server component. You can add and remove identities (as well as create new identities and mappings) as needed. For more information, see Configure security for a PI AF server.

4. Click **OK**.

Manage identities

Use the **Identities** tab to review identities that are currently assigned to the PI AF server. You can also add, remove, and edit identities.

Procedure

1. On the PI System Explorer toolbar, click .
2. In the PI AF Server Properties window, click the **Identities** tab.
3. Review the alphabetized list of identities. Click a column heading to change the sort order. Enabled identities are indicated as **True**, whereas disabled identities are grayed out and indicated as **False**. The built-in identities **Administrator** and **World** are read only.
4. You can choose from the following actions:

To ...	Do this ...
Disable an identity	a. Double-click an enabled identity on the list. b. In the Identity Properties window, clear the **Identity is Enabled** checkbox. c. Click **OK**.
Enable an identity	a. Double-click a disabled identity on the list. b. In the Identity Properties window, select the **Identity is Enabled** checkbox. c. Click **OK**.
Create a new identity	a. Right-click and select **New Identity**. b. In the Identity Properties window, enter a name and description for the identity. c. Optional. Click the **Mappings** tab and click **Add**. For more information, see Add a mapping to an identity.

Manage security settings for identities	a. Right-click a column title or below the list and click **Security**. b. In the Security Configuration window, you can review or modify permissions for a listed identity, as well as add or remove identities as needed. For more information, see Configure security for a PI AF server.
Delete an identity	a. Right-click an identity on the list and click **Delete**. b. In the Delete confirmation window, click **Yes**.
Rename an identity	a. Right-click an identity on the list and click **Rename**. b. Enter a new identity name. c. Right-click and click **Check In** to apply the change.

Manage mappings

Use the **Mappings** tab to review mappings that are currently set up on the PI AF server. You can also add, remove, and edit mappings.

Procedure

1. On the PI System Explorer toolbar, click .
2. In the PI AF Server Properties window, click the **Mappings** tab.
3. Review the alphabetized list of mappings. If necessary, scroll rightward to see mapped identities and Windows accounts.
4. You can choose from the following actions:

To ...	Do this ...
Modify the properties of a mapping	a. Double-click a mapping on the list. b. In the Mapping Properties window, you can edit the name and description as needed, as well as select a different identity from the **Identity** drop-down list. c. Click **OK**.
Create a new mapping	See Add a mapping to an identity.
Delete a mapping for an identity	a. Right-click a mapping on the list and click **Delete**. b. In the Delete confirmation window, click **Yes**. **Note:** If your user account is associated with the mapping you are deleting, a warning that you might be locked out is displayed.

Rename a mapping	a. Right-click a mapping on the list and click **Rename**. b. Enter a new mapping name. c. Right-click and click **Check In** to apply the change.
Manage security settings for mappings	a. Right-click a column title or below the list and click **Security**. b. In the Security Configuration window, you can review or modify permissions for a listed identity, as well as add or remove identities as needed. For more information, see Configure security for a PI AF server.

Add a mapping to an identity

When you add a mapping for an identity, you need to specify the Windows account to be mapped and the name of the mapping.

Before you start

The security identifier (SID) for the account is generated automatically and cannot be changed while the mapping is being created.

Procedure

1. In the **Account** field of the Mapping Properties window, enter one of the following object types:
   - User
   - Group
   - Built-in security principal
   - Service account
   - Computer name

   You can also click and enter search criteria in the Select User, Computer, Service Account, or Group window to select the object type, location, and name. The available search locations are the local computer hosting the connected PI AF server and active directory nodes that the user can access.

   Note:
   You can also enter a SID directly in the **Account** field. When the entry is validated, a domain identifier is prepended if the account is a domain account. If the account is a local account on the PI System's host machine, the fully qualified domain name is prepended.

2. To validate your entry in the **Account** field, press Tab or click in another field.

   The **Account SID** field displays the SID of the account, and the **Name** field defaults to displaying the account name.

3. In the **Name** field, modify the default account name as needed.

4. Text you enter in the **Description** field is displayed in the PI AF Server Properties window, so you can use this field to differentiate between mappings for any given identity (since in PI AF a single user can be mapped to multiple identities).
5. If an identity has not been selected already, select the identity you wish to associate with the mapping from the **Identity** drop-down list.
6. Click **OK**.

View PI AF server object counts

You can view object counts for the PI AF server to which you are currently connected, such as the number of databases, elements, element templates, event frames, and so on.

Procedure

1. In PI System Explorer, click **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Counts** tab.

PI Data Archive connections

An administrator needs to select a default PI Data Archive, if one was not selected during PI AF Client installation.

Topics in this section

- Connect to PI Data Archive
- Add PI Data Archive servers to connected server lists
- View buffering status for connected PI Data Archive servers

Connect to PI Data Archive

Use the Servers window to review PI Data Archive known server connections and to connect to different servers as needed.

Procedure

1. Click **File > Connections**.

   In the Servers window, you see a list of known servers, identified by the following icons.

	A disconnected PI AF server in the Known Server Table.
	A connected PI AF server.
	The default connected PI AF server.
	A PI Data Archive in the Known Server Table.
	The default connected PI Data Archive.

> **Note:**
> A warning icon ( ) beside a server indicates that PI Data Archive is running an unsupported version, such as version 3.4.375 or 3.4.370. The PI AF client cannot connect to a PI Data Archive server that is running software earlier than version 3.4.370.

2. To connect to a PI Data Archive, choose from the following actions:

To ...	Do this ...
Connect to a different PI Data Archive	Select a server in the list and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server in the list and click **Connect As**. b. In the Connect to PI Data Archive *server* window, enter a Windows user account name and password. c. Click **OK**.
Connect to a server that is not displayed on the Known Servers Table	Click **Add Data Server**. For more information, see Add PI Data Archive servers to connected server lists.
Set a different server as the default PI Data Archive	Right-click a server in the list and click **Set as Default Data Archive**. A icon is displayed beside the server that is now the default.

3. Click **Close**.

Topics in this section

- PI Data Archive connection problems
- Configure clients to allow explicit login prompts

PI Data Archive connection problems

Windows Authentication only uses identity mappings and does not utilize trusts defined with windows credentials for authentication or any other trusts that may be configured. When connecting to PI Data Archive, error messages might appear if there is no PI mapping configured and one of the following conditions exists:

- You are using a PI username and password to login to PI Data Archive.
- You are using the default user with a blank password to login to PI Data Archive.

These errors occur because PI AF 2012 and later has more stringent security settings than earlier versions of PI AF. Clients that connect to PI Data Archive through PI AF 2012 and later are subject to the following behaviors:

- The default behavior for connections is to not prompt the user for an explicit login unless the **Allow login prompt** setting is configured. This setting controls what happens when PI Data Archive authentication fails. If the **Allow login prompt** setting is enabled, a window opens prompting the user for a username and password. If the setting is off, an error message is displayed. The setting is specific to each client computer. For more information, see Configure clients to allow explicit login prompts.
- Default user connections are no longer supported.

To resolve error conditions, configure PI mappings for users connecting to PI Data Archive. For more information, see the PI System Management Tools topic "Identities, Users, and Groups" in Live Library (https://livelibrary.osisoft.com).

PI Data Archive connection error messages

Error Message	Connect Failure Conditions
Insufficient privilege to access PI Data Archive. No Access - Secure Object Default User connections to PI Data Archive are currently enabled from this client but are not supported from PI AF.	Default User connections are enabled in PI SDK.
Insufficient privilege to access PI Data Archive. No supported authentication methods are enabled. No Access - Secure Object Default User connections to PI Data Archive are currently enabled from this client but are not supported from PI AF.	Default User connections are the only authentication method enabled in PI SDK.
Insufficient privilege to access PI Data Archive. No Access - Secure Object The explicit login prompt is not configured on this computer and therefore not allowed.	Explicit logins setting was not set in registry.

Configure clients to allow explicit login prompts

If all configured protocols fail, use the PI SDK Utility to allow a login prompt to be displayed.

Procedure

1. Open **PISDKUtility** from the Windows Start menu.
2. Click **Connections** > **Options**.
3. In the Connection Options window, select the **Allow login prompt (if all configured protocols fail)** check box.
4. Click **OK**.
5. Click **File** > **Exit PISDKUtility**.

Add PI Data Archive servers to connected server lists

If the PI Data Archive server you want to connect to is not currently displayed on the Known Servers Table, add it in the PI Data Archive Properties window.

Procedure

1. Click **File** > **Connections**.
2. In the Servers window , click **Add Data Server**.
3. In the PI Data Archive Properties window, enter properties as needed.

4. Optional. You can change the default name in the **Name** field:
   a. Click **Rename**.
   b. In the **New Name** field in the New PI Data Archive Name window, enter a name for the new PI Data Archive. The name does not need to match the host name.
   c. Click **OK**.
5. Enter the host name in the **Host** field. You can enter:
   - A fully qualified domain name.
   - A server name.
   - An IP address. An IPv6 address must be enclosed in brackets [ ].
6. Unless your particular application requires a different port, accept the default value in the **Port** field.

   Note:
   You can modify the host name and port only when disconnected from the server.
7. The default values for the **Connection Timeout** and **Data Timeout** fields are acceptable in most cases. If you experience connection timeouts when connecting to PI Data Archive from PI System Explorer, increase the time in the **Connection Timeout** field. If you experience timeout errors while accessing PI Data Archive data, increase the time in the **Data Timeout** field.
8. Optional. Enter an alias name in the **Aliases** field.

   Aliases are alternate names that can be used for PI Data Archive. PI Data Archive aliases are stored locally on the client only.
9. Click **Connect** to connect to PI Data Archive.

   Note:
   The **ID**, **Time Zone** and **Version** fields are not editable. **ID** is the PI Data Archive ID, **Time Zone** is the local time zone of the PI Data Archive and **Version** is the PI Data Archive version.
10. Click **OK**.

View buffering status for connected PI Data Archive servers

If PI Buffer Subsystem 4.3 or later is installed and running, you can view the buffer status of a connected PI Data Archive server.

Procedure

1. Click **File** > **Connections**.

   In the Servers window, each PI Data Archive server for which a connection is configured displays a buffering status in the **Buffer Status** column.

   Note:
   PI Buffer Subsystem versions 3.4.380 and earlier display a status of **Unknown**.
2. To view more details and manage buffering for a specific PI Data Archive, click **Buffering Manager** to open the Buffering Manager tool.

> **Note:**
> You can also open Buffering Manager from the PI System Explorer **Tools** menu.

PI AF database connections

PI AF stores the asset framework objects (elements, templates, and so on) in PI AF databases. You can have multiple databases in PI AF, although you can connect to only one at a time. You typically work with PI AF databases in PI System Explorer or in PI Builder. When you start PI System Explorer, it connects to the default PI AF database. If no databases are defined, PI System Explorer prompts you to create a new database.

> **Note:**
> The **Configuration** database is used by PI AF clients, such as PI Web API, as they interact with PI AF. Do not use this database for your own application data.

Videos

For information on how to create PI AF databases, watch this video:

https://www.youtube.com/watch?v=P7Zopif-j-c&rel=0

For information on how to work with multiple PI AF databases, watch this video:

https://www.youtube.com/watch?v=CN0R7cW3BUc&rel=0

For information on the configuration database, watch this video:

https://www.youtube.com/watch?v=bfTnAeSinmo&rel=0

Topics in this section

- Create PI AF databases
- View or edit properties of PI AF databases
- Connect to a database on a different PI AF server
- Set the default database
- Rename databases
- Search for PI AF databases
- Refresh the list of databases
- Database deletion

Create PI AF databases

You can create an empty PI AF database with no pre-configured content, or you can load a library that has been saved on a connected PI AF server.

Procedure

1. Choose one of the following actions:

- Press Ctrl+O.
- On the PI System Explorer toolbar, click Database.
- Click **File** > **Database**.

2. From the **Asset server** drop-down list in the Select Database window, select a connected server on which you have administrator access rights to create a database. Alternatively, click ... and select a server from the PI AF Servers window.
3. Click **New Database**.
4. In the Database Properties window, enter a unique name in the **Name** field.
5. Optional. Enter a description in the **Description** field.
6. Choose from the following actions:

To ...	Do this ...
Create a new database with no configured content	Leave the **Load Library** field (if displayed) set to <None>.
Create a database with configured objects from a saved library	Select a library from the **Load Library** drop-down list. For more information, see PI AF saved libraries.
Create extended properties	See Storage of application-specific information.
Set up access permissions for the new database	See Configure security for a single PI AF database.

7. Click **OK**. The new database is added to the **Database** list in the Select Database window.

View or edit properties of PI AF databases

Use the Database Properties window to check how many objects exist in each collection type in a PI AF database, as well as to review or add extended properties and access permissions.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Properties**.
3. In the Database Properties window, choose from the following actions:

To ...	Do this ...
Review the number of objects in the database	Click the **Counts** tab.
Review or modify the database name and description	On the **General** tab, make changes as needed in the **Name** and **Description** fields.
Review or modify extended properties	a. On the **General** tab, click **Extended Properties**. b. In the Extended Properties window, make changes as needed in the **Name** and **Value** fields. To create a new property, click **New Extended Property**. For more information, see Create extended properties.

Review or modify database security	a. On the **General** tab, click **Security**. b. In the Security Configuration window, make changes as needed. For more information, see Configure security for a single PI AF database.
Review the properties of the connected server	a. On the **General** tab, click . b. In the PI AF Server Properties window, review properties on the **General** tab, as described in View general properties of the connected PI AF server.
Load a saved library	a. On the **General** tab, select a saved library from the **Load Library** list. For more information on saved libraries, see PI AF saved libraries. b. Click **Apply**.

4. Click **OK**.

Connect to a database on a different PI AF server

Use the PI AF Servers window to locate and connect to the PI AF server that contains the database you want to work with.

Procedure

1. In the Select Database window, click .
2. In the PI AF Servers window, review the list of servers. Note that the default database on each connected server is also displayed.
3. Choose from the following actions.

To ...	Do this ...
Select the server where the database is located	Double-click on a connected ( ) server. The PI AF Servers window closes and the selected server is displayed in the **Asset server** field.
Check the properties of a server	Select a server and click **Properties**.
Connect to a server	Right-click a server that is not connected ( ) and click **Connect**.
Connect to a server as a different user (for example, from a shared computer)	a. Right-click a server that is not connected and click **Connect As**. b. In the Connect to PI AF Server *server* window, enter a Windows user account name and password. c. Click **OK**.

Review access permissions for a connected server	a. Right-click a connected server and click **Security**. b. In the Security Configuration window, check the settings for the listed identities, as needed. **Note:** If you have administrative privileges on the server, you can also modify the security configuration. See Configure security for new PI AF databases.

Set the default database

The default database is displayed when PI System Explorer opens for the first time. After the first time you run PI System Explorer, it will display whatever database was open when you last closed PI System Explorer.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Set as Default Database**. A check mark icon ( ) is displayed over the database icon that is now the default.
3. Click **OK**.

Rename databases

You can change the name of a PI AF database as needed.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click a database and click **Rename**.
3. Enter a valid name for the database. For rules on object names, see Valid characters in PI AF object names.
4. Click **OK**.

Search for PI AF databases

To locate a specific PI AF database on a selected PI AF server, use the Search feature.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, begin typing the name of a database in the **Databases** field.

   The **Search** icon changes to the **Searching** icon . Results filter corresponding to how many letters you type.

3. Select a database in the list and click **OK**. The window closes, and you will be working in the selected database.
4. To return to the full list of available databases, press ESC. The window closes.

Refresh the list of databases

When other users add databases to the PI AF server, those databases might not be displayed until you refresh.

Procedure

1. On the PI System Explorer toolbar, click Database.
2. In the Select Database window, right-click in the list of databases and click **Refresh**.

Database deletion

When you delete a database, all information contained within that database is removed. Before you delete a database, therefore, ensure that you have selected the database that you want to delete.

> **Caution:**
> You cannot undo a deletion. However, you can recover data by restoring your last SQL Server backup.

PI AF and PI Data Archive collective connections

An administrator needs to configure connection preferences if a PI AF collective or PI Data Archive collective is being set up.

Topics in this section

- Manage connection preferences for PI System Explorer
- Switch collective members
- Switch to a specific collective member

Manage connection preferences for PI System Explorer

Use the **Server Options** window to define how PI System Explorer should connect to a PI AF collective or PI Data Archive collective, whether login prompts are allowed, and if you want servers to be added to the Known Servers Table automatically.

Procedure

1. In PI System Explorer, choose **Tools** > **Options** > **Server Options**.
2. In the **PI AF Server Connection Settings in PI System Explorer** section, define how PI System Explorer should connect to the PI AF server:

Setting	Description
Connection preference	Use this preference to influence the selection of a collective member when a connection is made to a PI AF collective. ◦ **Prefer Primary** PI System Explorer attempts to connect with the primary server in the collective, and if it is not available, uses the individual priority settings of the collective member to influence selection of the server connection. This is the default setting. ◦ **Require Primary** PI System Explorer is required to connect with the primary server in the collective. If the primary server is not available, the connection fails. ◦ **Any** PI System Explorer can connect with any available member of the collective, and uses the individual priority settings of the collective member to influence selection of the server connection.
Server name resolution behavior	Use this preference to inform PI System Explorer what action to perform when it attempts to communicate with one or more PI AF servers that are not in the Known Servers Table. ◦ **Auto Add** Adds a server to the Known Servers Table automatically if the computer name is valid and PI System Explorer can find the server's computer name on the network. ◦ **Auto Add If Resolvable** Adds a server to the Known Servers Table only when the server name is successfully resolved by a DNS server. ◦ **Do Not Auto Add** Makes no attempt to find a server if it cannot be found in the existing collective.
Allow login prompt	Select this option if you want a login prompt to appear when a connection to the server fails because of a security exception.

3. In the **PI Data Archive Connection Settings in PI System Explorer** section, define how PI System Explorer should connect to PI Data Archive:

Setting	Description
Connection preference	This preference is used to influence the selection of a collective member when a connection is made to a PI Data Archive collective. ◦ **Prefer Primary** PI System Explorer attempts to connect with the primary server in the collective, and if it is not available, uses the individual priority settings of the collective member to influence selection of the server connection. This is the default setting. ◦ **Require Primary** PI System Explorer is required to connect with the primary server in the collective. If the primary server is not available, the connection fails. ◦ **Any** PI System Explorer can connect with any available member of the collective, and uses the individual priority settings of the collective member to influence selection of the server connection.
Server name resolution behavior	Use this preference to inform PI System Explorer what action to perform when it attempts to communicate with one or more PI Data Archive servers that are not in the Known Servers Table. ◦ **Auto Add** Adds a server to the Known Servers Table automatically if the computer name is valid and PI System Explorer can find the server's computer name on the network. ◦ **Auto Add If Resolvable** Adds a server to the Known Servers Table only when the server name is successfully resolved by a DNS server. ◦ **Do Not Auto Add** Makes no attempt to find a server if it cannot be found in the existing collective.

4. In **Protocols** in the **PI Data Archive Connection Settings** section, set the authentication protocols and the order in which they are used when connections are attempted to known servers.

   PI System Explorer attempts to connect using the first protocol listed. If the first attempt fails, it continues attempting to connect using the other protocol in the list (if selected). If attempts using all protocols fail and **Allow login prompt** is selected, the PI Data Archive **Login** window is displayed.

   a. To establish the order used when connecting to a server, use the arrows button to move protocols.

   b. Select the **Allow login prompt** option if you want a login prompt to be displayed when a connection to the server fails because of a security exception.

5. Click **OK**.

Switch collective members

When you connect to a PI AF collective, PI AF automatically connects you to the collective member with the highest priority (lowest number). You can switch the connection to the next member in the collective list. The next member in the list is determined by members' assigned priority.

Procedure

1. In PI System Explorer, click **File** > **Connections**.
2. Right-click the collective and click **Switch Collective Member**.

Switch to a specific collective member

When you connect to a PI AF collective, you are automatically connected to the collective member with the highest priority (lowest number). You can switch to a specific member of the collective.

Procedure

1. In PI System Explorer, select **File** > **Connections**.
2. Right-click the collective and click **Connect to Collective Member**.
3. In the Choose Collective Member window, select the collective member from the **Collective Member** list to which you want to connect.
4. Click **OK**. You are now connected to the selected collective member.

Database import and export

You can export a database from PI System Explorer in XML and then restore it from that XML file by importing it back to PI AF.

To export the database without having to manage XML files, you export the database as a library, as described in Save databases as libraries.

Videos

For information on how to import and export databases with PI System Explorer, watch this video:

https://www.youtube.com/watch?v=ClGTy_lkoTg&rel=0

For information on how to import and export databases with import and export utilities, watch this video:

https://www.youtube.com/watch?v=zc5pcH1XvMo&rel=0

Topics in this section

- Export a database to XML
- Restore databases from exported XML files
- Export of library objects to another database
- AFExport utility
- AFImport utility

Export a database to XML

This procedure archives PI AF databases into an XML file that you can later restore.

> **Note:**
> You can also export PI AF objects from a command line utility. See AFExport utility.

Procedure

1. Click **File** > **Export to File**.
2. In the Export to File window, select desired export options (XML export options).
3. Click **OK**.

   PI System Explorer exports the current database into an XML file.

Topics in this section

- XML export options
- All Referenced Objects
- File format for export and restore

XML export options

The Export to File window contains the following export options.

Export option	Result
Include All Referenced Objects	Causes dependent references to be exported, as detailed in the All Referenced Objects table. Use this option to facilitate moving objects between different PI AF databases.
Include Security Settings	Causes the security settings of individual objects to be exported. This option increases the amount of time required for the export and subsequent import operations. You must have Administrative privileges to import objects that have the security setting specified.
Flatten XML	Exports hierarchical objects in a flat format. Exporting in a flat style can make editing in some tools simpler. Hierarchical objects that will be exported flat are attributes, attribute templates, elements, and event frames.
Simplify Configuration Strings	Removes PI Data Archive and tag identifiers from configuration strings for PI point data references. Additionally, substitution parameters will be resolved. **Note:** This option slows the export process because it requires evaluation of the saved configuration strings.
Include Default Values	Includes the default values assigned to objects. Without this option, a property that has not been changed from its default setting is not included in the output. This saves considerable space and time when you are exporting large amounts of data.
Include Unique IDs	Includes the unique ID of each object in the export. By including this option, you can rename existing items during an import to the same database. Unless rename is required, it is more efficient to leave this option turned off.
Library Objects Only	Disables the event frames, transfers, and cases options.
Include Event Frames, Transfers, and Cases	Exports event frames, transfers, and cases.

All Referenced Objects

The following table lists the objects that are exported when you select **Include All Referenced Objects**.

XML object type	Included reference
`AFAnalysis`	`AFAnalysisCategories` referenced
	`AFAnalysisTemplate` in `Template`
	`AFElementTemplate` in `CaseTemplate`
	`AFElement` in `Target`
	`AFSecurityIdentities` from `SecurityAccessControl` (if **Include Security Settings** is also selected)

XML object type	Included reference
AFAnalysisTemplate	AFAnalysisCategories referenced
	AFElementTemplate in CaseTemplate
	AFElement in Target
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFAttribute	Referenced AFEnumerationSets in ValueTypeQualifier when not from template
AFAttributeTemplate	Referenced AFEnumerationSets in ValueTypeQualifier
AFCase	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	All child AFElements
	All AFConnections (not just those added or removed)
AFContact	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFDatabase	UOMDatabase
	AFContacts referenced
	AFNotificationContactTemplates referenced
	AFSecurityIdentities (if **Include Security Settings** is also selected)
AFElement and AFModel	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFReferenceTypes to referenced children
	Same as above for all child elements
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFElementTemplate	Base AFElementTemplates
	AFElementTemplates referenced in AFPort.AllowedElementTemplate
	AFReferenceTypes that specifically reference this template
	AFElementCategories referenced
	AFEnumerationSets referenced in attribute templates
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFEnumerationSet	AllReferences adds AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFEventFrame	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFReferenceType from parents
	AFElements referenced
	Same as above for all child event frames
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)

XML object type	Included reference
AFModelAnalysis	AFAnalysisCategories referenced
	AFElementTemplate in Template
	AFElementTemplate in CaseTemplate
	AFElement in Target
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotificationTemplate	AFNotificationContacts
	AFAnalysisTemplate
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotification	AFNotificationTemplate in Template
	AFNotificationContacts
	AFAnalysis
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFNotificationContact	AFNotificationContactTemplate
	Child AFNoticationContacts
	AFContact referenced
AFNotificationContactTemplate	Child AFNotificationContactTemplates
	AFContacts referenced
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFDeliveryFormat	DeliveryChannel
AFNotificationRule	AFNotificationRuleTemplate
	AFNotificationContactTemplate (via Subscribers)
	Child AFNoticationContacts
	AFContact
	AFDeliveryFormat
AFNotificationRuleSubscriber	AFNotificationContactTemplate
	Child AFNoticationContacts
	AFContact
AFNotificationRuleTemplate	AFNotificationContactTemplate (via Subscribers)
	Child AFNoticationContacts
	AFContact
	AFDeliveryFormat
AFReferenceType	AFReferenceTypeCategories referenced
	AFElementTemplates from AllowedParentElementTemplate and AllowedChildElementTemplate
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFSecurityIndentity	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)

XML object type	Included reference
AFSecurityMapping	AFSecurityIdentity mapped to
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTable	AFTableConnection
	AFTableCategories referenced
	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTableConnection	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
AFTransfer	AFElementTemplate in Template
	AFElementCategories referenced
	AFEnumerationSets referenced in attributes without templates
	AFElements referenced in Source and Destination
UOMClass	UOMClass for exported class and classes referenced by UOMs in UOMClass. For example, when the Area class is exported, the Length class is included because Length UOMs are referenced by many Area UOMs. **Note:** UOMClass export does not include security settings, which are only exported with UOMDatabase.
	UOMGroup (if any of the UOMs in the exported or referenced classes have an assigned UOMGroup).
	UOM for all UOMs belonging to the exported UOMClass.
UOMDatabase	AFSecurityIdentities from SecurityAccessControl (if **Include Security Settings** is also selected)
	UOMGroup. Group mappings are defined under each UOM.
	UOMClass
	UOM

File format for export and restore

The Import/Export XML format is described in a schema file located in the PIPC\AF installation directory, OSIsoft.AF.xsd.

Many of the AF Objects support an operation attribute in the XML that allows for deletion. Similarly, you can execute the **Auto Point Config** option selectively on elements using this same technique. Example:

```
...
<AFElement operation=delete> <Name>ElementToDelete</Name> </AFElement>
...
```

Elements and Attributes can be imported using a flat list, in which the relative path of the element or attribute is included in the name field. Example:

```
...
<AFElement> <Name>RootElement</Name> ... </AFElement>
<AFElement> <Name>RootElement\ChildElement1<\Name> ... </AFElement>
```

```
<AFElement> <Name>RootElement\ChildElement2<\Name> ... </AFElement>
...
```

Restore databases from exported XML files

You can restore a database that has been exported to XML by importing objects from the XML file.

You can also import PI AF objects from a command line utility. See AFImport utility.

Before you start

OSIsoft recommends that you do not use the **Allow Update** option when importing objects with duplicate names, such as event frames, into an empty PI AF database. Otherwise, problems can occur. However, you may then receive a message that objects, such as units of measure, already exist on the PI AF server. The import does not attempt to make any changes to existing objects, which means that if there are differences between objects' properties in the XML file and the destination system, they are not resolved.

Procedure

1. Click **File** > **Import from File**.
2. In the Import from File window, select the XML file that contains the data and select the appropriate options.

Import option	Description
Allow Create	Allows new objects to be created. If you want to update existing items, clear this option to prevent accidental creation of new objects.
Allow Update	Allows existing objects to be updated. If you only want to add new items, clear this option to prevent existing objects from being accidentally overwritten.
Automatic Check In	Allows the import operation to automatically check in objects as the import proceeds. This reduces the maximum memory requirements of the operation. Use this option when you are importing a large number of objects.
Create or Update PI Points	Causes any attributes with a PI point configuration specified in the XML file to be updated if they already exist, or created if they do not exist. This option invokes the appropriate Data Reference **CreateConfig** option, which creates or updates PI points, as well as resolving their substitution parameters and setting server and point IDs. The performance of the import operation is negatively affected when this option is selected. **Note:** PI points are not created unless the **Tag Creation** check box is selected in the PI Point Data Reference window.

Preserve Unique IDs	Retains any unique IDs that exist in the source XML. This option is only valid when the **Allow Create** and **Automatic Check In** options are also selected. The performance of the import operation is negatively affected when this option is selected. There is no guarantee that PI AF will preserve unique IDs of attributes derived from templates because unique IDs of such attributes are generated based on the template's unique ID. **Note:** Because PI AF servers maintain a cache of `rowids` to `ids`, the import with **Preserve Unique IDs** option affects load-balanced PI AF servers differently from non-load balanced PI AF servers. In a non-load balanced scenario, removed items will cause the cache of `rowids` to `ids` to be updated. However, in a load-balanced scenario, the non-connected server will need to be restarted to have the cache updated after items are removed, and before the import with **Preserve Unique IDs** option is used to restore items to the same system.

Export of library objects to another database

You can export library objects (templates, formulas, UOM, and so on) from one database into another.

Topics in this section

- PI AF saved libraries
- Save databases as libraries
- Load database libraries
- Review properties of loaded libraries

PI AF saved libraries

A PI AF saved library provides a collection of application- or domain-specific objects that you can import into a PI AF database. Saved libraries typically include categories, element templates, enumeration sets, reference types, tables, as well as the unit-of-measure database, which is always included. You also have the option to include other objects, such as elements and notifications.

Note:
Libraries are stored as XML files in the PI AF SQL Server database (PIFD), where they are easily accessible to other users. By contrast, the **Export to File** option enables you to export an entire database of objects as an XML file to your computer or network. The exported file can be imported to a different server altogether.

Save databases as libraries

You save database objects as a library when you wish to make those objects available for import to a different database on the PI AF server.

Procedure

1. Click **File** > **Save as Library**.
2. In the Save Database as Library window, enter a name in the **Library Name** field. The library name does not need to be unique if you plan to overwrite an existing library.
3. Optional. Enter a description of the library content in the **Description** field.
4. To replace an existing library, select the **Replace existing Library without prompting** check box.
5. To include objects such as elements and notifications in the library, select the **Include non-library objects** check box.
6. Click **OK**.

Load database libraries

You load a database library from a set of libraries that have been previously saved in the PI AF database.

Procedure

1. Click Database.
2. In the Select Database window, select the database into which you want to load a library, and click **OK**.
3. Click **File** > **Load Library**.
4. In the Load Library into Database window, select a library from the list and click **OK**.

Results

The objects in the loaded library populate the current database. To confirm how many objects are loaded, view the **Count** tab in the Database Properties window, as described in View or edit properties of PI AF databases.

Review properties of loaded libraries

You view the properties of libraries that have been loaded on the PI AF server in the PI AF Server Properties window. You can change a library name and description.

Procedure

1. Click **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Libraries** tab.

   The libraries that are currently installed are listed.
3. Right-click the library you want to review and click **Properties**.

4. Optional. In the Library Properties window, you can change the library name and description as needed.
5. Click **OK**.

AFExport utility

The AFExport utility is a command line application that you can use to archive PI AF databases into an XML format that you can restore later. Use this utility to archive elements, templates, event frames, transfers, and other objects from a PI AF database. You can also export collections of PI AF objects from the PI System Explorer. See Export a database to XML.

The `AFExport.exe` utility is located in the `\PIPC\AF` folder.

To run the PI AF Export utility you open a command window and navigate to the `PIPC\AF` folder. Use the following syntax: `AFExport.exe` and choose from the parameters listed in AFExport utility parameters. To display all parameters, type:

```
AFExport /?
```

Topics in this section

- Guidelines for exporting collections
- AFExport utility parameters
- Sample export paths

Guidelines for exporting collections

You should follow these guidelines as you prepare to export a collection:

- Specify collections by their PI AF SDK property name, followed by "[ ]".
- Identify individual members of a collection by placing their name within the brackets.
- If no collection name is specified, the default child collection for that location in the path is assumed.
- The default child collection of a PI AF server is `Databases`.
- The default child collection of a database is `Elements`.
- The default child collection of an element is `Elements`.
- The default child collection of UOM database is `UOM`.
- A vertical bar (|) indicates an attributes or attribute templates collection.

AFExport utility parameters

The **`AFExport`** utility supports the following parameters.

Parameter	Short form	Description
Path Additional details and examples follow this table.		Path to the object that you wish to export. Typically of the form: *****afserver* *****database*. Use '`.`' to export the default database. See Sample export paths.

Parameter	Short form	Description
/File:*string*	**/F**	Specify an output file. If not specified, then the output is streamed to the console.
/StartTime:*string*	**/T**	Specify a start time to cause event frames, transfers, and cases to be exported that are between the start and end times specified.
/EndTime:*string*	**/E**	Specify an end time when exporting event frames, transfers, and cases.
/AllReferences[-]	**/A**	Export any referenced object from the specified PI AF object. This parameter cannot be used with the **No References** parameter.
/NoReferences[-]	**/N**	Do not export any referenced objects, including child objects. This parameter cannot be used with the **All References** parameter.
/DefaultValues[-]	**/D**	When this parameter is not specified, then properties that are set to the default values are not exported, resulting in a substantially smaller export file. Importing a file without default values over existing objects may result in values not being reset to their defaults.
/Library[-]	**/L**	Export only the library objects from a database. Library objects include all categories, templates, enumeration sets, reference types, and tables.
/Security[-]	**/Y**	Export security information. Exporting security information will increase both export time and subsequent import times.
/SimplifiedConfigStrings[-]	**/Sc**	Export configuration strings for attribute templates or attributes derived from attribute templates without substituting parameters and do not export UniqueIDs or point identifiers in the configuration strings.
/UniqueIDs[-]	**/U**	Export unique IDs of all objects. This allows the renaming of objects when imported back into the same database. This parameter increases the size of the output file and may increase the amount of time required to import.
/Silent[-]	**/S**	Silent mode. Prevents informational messages from being displayed. If no output file is specified, this option is automatically chosen.

Parameter	Short form	Description
/Summary[-]	**/M**	Summary mode. Output only minimal information on progress. This parameter is not valid with the **Silent** mode or if no output file is specified.
/User:*string*	**/user**	Use to specify a different Windows user account to connect to the PI AF server.
/Password:*string*	**/pw**	If a user name is specified, specify the network credentials password.
/Version	**/V**	Use to display version information. All other parameters are ignored.
/Flat[-]	**/flat**	Export hierarchical object in a flat format. Exporting in a flat style can make editing in some tools simpler. Hierarchical objects that will be exported flat are attributes, attribute templates, elements, and event frames.
/Paste[-]	**/Pa**	Paste operation behaves as a typical copy/paste.
/?, **/help**		Prints the contents of this table.
@*file*		Use the specified file to provide additional input arguments. The file should contain one argument per line. Comment lines start with the **#** character.

Sample export paths

The following table contains sample syntax to export different PI AF database components.

To export ...	Use this syntax ...
The default database	`.`
The database `MyDatabase` on the default PI AF server	`\\.\MyDatabase`
The element `MyElement` in the database `MyDatabase` on the PI AF server named `MyAFServer`	`\\MyAFServer\MyDatabase\MyElement`
All element templates in the database `MyDatabase`	`\\.\MyDatabase\ElementTemplates[]`
Element template `T1` in the database `MyDatabase`	`\\.\MyDatabase\ElementTemplates[T1]`
All enumeration sets in the `MyAFServer` default database	`\\MyAFServer\.\EnumerationSets[]`
All attributes of `MyChildElement` in the `MyAFServer` default database	`\\MyAFServer\MyElement\MyChildElement\Attributes[]`
The UOM database (`UOMDatabase` is case sensitive)	`\\.\UOMDatabase`
All tables in the `MyAFServer` default database	`\\MyAFServer\.\Tables[]`

AFImport utility

The **AFImport** utility is a command line application that you can use to restore PI AF objects into a database. You can also use **Import from File** in PI System Explorer to restore database objects. See Restore databases from exported XML files.

The **AFImport.exe** utility is located in the \PIPC\AF folder.

To run the **AFImport** utility, you open a command window and navigate to the PIPC\AF folder. Use the following syntax: AFImport.exe and choose from the parameters listed in AFImport utility parameters. To display all parameters, type:

```
AFImport /?
```

AFImport utility parameters

The **AFImport** utility supports the following parameters.

Parameter	Short form	Description
Path		Path to the object into which you want to import. Typically of the form: *****afserver**database*. Use '.' to import into the default database.
/File:*string*	**/F**	Specify an input file. If not specified, the import operation reads from the standard input.
/AutoCheckIn[-]	**/A**	Automatically check in changes during an import operation. Default value: True
/Create[-]	**/C**	Allow import operation to create new objects. Default value: True
/Update[-]	**/U**	Allow import operation to update existing objects. Default value: True
/CreateCategories[-]	**/CC**	Create categories that do not exist but are referenced by objects in the input. Default value: True

Parameter	Short form	Description
/PreserveUniqueIDs[-]	**/Pid**	Retain unique IDs that exist in the source XML. This option is only valid when **/Create** and **/AutoCheckIn** are also specified. This option can significantly affect import performance. There is no guarantee that PI AF will preserve unique IDs of attributes derived from templates because unique IDs of such attributes are generated based on the template's unique ID. **Note:** Because PI AF servers maintain a cache of `rowids` to `ids`, the import with **Preserve Unique IDs** option affects load-balanced PI AF servers differently from non-load balanced PI AF servers. In a non-load balanced scenario, removed items will cause the cache of `rowids` to `ids` to be updated. However, in a load-balanced scenario, the non-connected server will need to be restarted to have the cache updated after items are removed, and before the import with **Preserve Unique IDs** option is used to restore items to the same system.
/CreateUpdatePIPoints[-]	**/P**	Create or update PI point configuration for newly created elements. This option can significantly affect import performance.
/DisableConfigStringValidation[-]	**/D**	Disable the validation of configuration string settings for data references and delivery channels, which can improve the speed of the import operation. However, this will bypass looking up PI points, which corrects or adds server IDs and point IDs to configuration strings.
/GenerateUniqueNames[-]	**/G**	Generate unique names for objects when an object with the same name already exists.
/Paste[-]	**/Pa**	Paste operation behaves as if this is a copy/paste process. Depending on where the data is imported, this affects whether a new name is generated.

Parameter	Short form	Description
/ContinueOnError	**/CE**	Specify the behavior of the import process when a recoverable error occurs. The three options available are `Yes`, `No`, and `Prompt`. The default value is `Prompt`, unless running silently in which case the default will be to cancel the import process.
/Silent[-]	**/S**	Silent mode. Prevent informational messages from being displayed.
/Summary[-]	**/M**	Summary mode. Output only minimal information on progress.
/User:*string*	**/user**	Use to specify a different Windows user account to connect to the PI AF server.
/Password:*string*	**/pw**	If a user name is specified, specify the network credentials password.
/Version	**/V**	Display version information. All other parameters are ignored.
@*file*		Read response file for more options. The response file contains one parameter per line. Comment lines start with the **#** character.

Retrieval of asset information

PI System Explorer provides a variety of methods that you can use to locate and review the assets in your PI AF database, including a browser, search windows, trends to verify formulas and calculations, and version management. You can also review time series data for attributes that contain or reference such data.

Topics in this section

- Asset browsing
- Asset and asset data searches
- Trends in PI System Explorer
- Version management for equipment and processes
- Display of different object versions and obsolete objects
- View time series data

Asset browsing

You use the PI System Explorer browser to display asset objects in the PI AF database. The browser displays the following assets, depending on the asset type that you have selected in the navigator pane:

- **Elements**

  Displays element assets and element searches in a tree format. You can control how many elements are displayed per page.

- **Event Frames**

  Displays event frame and transfer search collections. Because asset models can comprise thousands of event frames or transfers, these objects are not displayed in a hierarchy in the browser. Instead, each has a search collection in the browser. Recent searches are also listed under the search collection.

- **Library**

  Displays object collections for **Templates** (for elements, event frames, models, and transfers), **Enumeration Sets**, **Reference Types**, **Tables**, **Table Connections**, and **Categories** (for analyses, attributes, elements, notification rules, reference types, and tables). To see all the objects of a particular type, you expand the collection for that type.

- **Unit of Measure**

  Displays all UOM classes in the Unit of Measure database. When you select a class, UOMs belonging to that class are listed in both the viewer and the conversion calculator.

- **Contacts**

  Displays a list of contact searches, escalation teams, groups, and stand-alone delivery endpoints for use with notifications.

Topics in this section

- Element browsing

- Configure browser page size
- Open additional browser windows

Element browsing

Elements are displayed as a tree in the browser. The structure of the element tree is different for each organization. The asset model designer chooses structure that is relevant to the users in the organization. In the following illustration elements are organized by plant location. If you were in charge of all the equipment for a particular plant, then such a model might make sense for you. A different model might be more useful for someone with a different role.

The element tree can include different sub trees that provide different context for the same assets. This allows users to find elements in the context that makes the most sense for the task at hand. For example, in addition to the organization by plant location shown above, you might also have an organization by equipment manufacturer, as shown in the following illustration.

The elements representing the pumps appear in both the Manufacturers and the Plant Location hierarchies. In the Manufacturer hierarchy, the pump elements are not new separate elements, rather they are references to the elements that already existed in the Plant Location tree. To indicate that they are element references, they are represented by the referenced element icon .

Configure browser page size

The PI System Explorer browser is paged. This means that it displays a certain number of objects at a time. If more objects exist, PI System Explorer displays them in another page. When this happens, you see <Previous> and <Next> in the browser.

By default, the maximum number of objects displayed in the browser is 1000. You can change that number.

Procedure

1. In PI System Explorer, select **Tools** > **Options**.
2. On the **General** tab, type the maximum number of objects that can be displayed in the browser in the **Maximum Tree Branch Page Size** field.
3. Click **OK**.

Open additional browser windows

Occasionally, you may find it convenient to open additional browser windows. For example, you can have the Unit of Measure database displayed in one browser instance while creating a new template in the Library browser.

Procedure

1. In the navigator pane, right-click the item you want to open in another window.
2. Select **Open in New Window**.
3. To move between windows, click the PI System Explorer icon in the Windows taskbar and select the thumbnail of the window you want.

Asset and asset data searches

PI System Explorer provides a variety of options for finding assets, asset data, and PI points.

- Find data for a specific piece of equipment. For example: PI point data, calculation results, maintenance information, and so on. For example, you can use PI System Explorer to find information about a specific tank at a specific plant.
- Find all equipment with specified attribute values or value ranges. For example, get a list of all tanks with temperature greater than 200 °F.

Topics in this section

- Quick search
- Search result paging
- Maximum query sizes
- Searches on a specified date
- Search for PI points
- Manage search results
- Search for elements
- Special characters in name searches
  You can use special characters when searching for objects by name.
- Search for attributes on elements
- Search for event frames

- Search for attributes on event frames
- Search for transfers
- Search for attributes on transfers

Quick search

You can use the quick search box at the right end of the PI System Explorer toolbar to find elements, event frames, element templates, or UOM (unit of measure) types.

Procedure

1. Make a selection in the navigator pane to set the context for quick search. For example, select **Event Frames** to set quick search to search for event frames.
2. In the quick search box, select a search constraint from **Actions** ▾, enter search criteria, and press Enter. Search results are displayed in the viewer, as well as in the browser.

> **Note:**
> Search results that are displayed in the browser remain there for easy access in the event you need them again. You can right-click and delete them from the browser; they are also removed when you exit PI System Explorer. You can right-click and save them to keep them until you delete them.

Search result paging

When you are searching large numbers of elements or event frames, you can use the **Set Maximum Query Size** option to return quick search results back in manageable chunks. This is called *paging*. Change the **Set Maximum Query Size** value to the number of objects you want to see in the search results at one time. If the search returns a greater number of objects than the **Set Maximum Query Size** value specifies, the results are paged.

For example, if the **Set Maximum Query Size** value is 100,000 the PI AF server attempts to find 100,000 matching objects before returning the results. You probably would not want to wait for the server to find that many matches, nor would you want to see that many matches in a single page of search results. If you change **Set Maximum Query Size** to 100, the PI AF server would send you the results after each 100 matches, and you would see 100 objects at a time in the search results.

Search result rules

- Only search results for elements, transfers, and event frames are paged.
- The **Set Maximum Query Size** setting applies only to quick searches of elements, transfers, or event frames. If you are searching other objects, the search ignores the **Set Maximum Query Size** value.
- In search windows, the **Results per Page** value overrides the **Set Maximum Query Size** value.

Maximum query sizes

A system default controls the number of objects that can be returned by a search query. Users can override the system default by specifying a different value in Element Search, Attribute Search, Event Frame Search, and Transfer Search windows.

System default

You set the **Maximum Query Size** option in **Tools** > **Options**. This sets the default value for PI System Explorer.

Custom sizes

You can override the system default for a specific search by setting a custom value for **Maximum results**. For event frames, you specify a custom value in the **Results per Page** field.

Searches on a specified date

You can conduct a search in PI System Explorer at a specified time and date, by entering that time setting for **Query Date**. PI System Explorer also displays object versions for that date. See Query Date and PI System Explorer time for information on what objects are affected by **Query Date**.

In addition, you can set a time or time range that applies only to displayed attribute values and leaves in place the **Query Date** setting. To do this, establish a time context in the Set Time Context window, as described in Set time context for displayed attribute values.

Search for PI points

Use the Tag Search window to retrieve PI points that match your search query criteria. In PI AF and PI Builder 2017 or later versions, you can create a search query based on both PI point attributes and PI point values.

> **Tip:**
> To create PI point data references directly from PI points retrieved from a PI point search, select **View** > **Palette** > **Tag Search**. For more information, see Create direct PI point data references from Tag Search results.

Procedure

1. In the navigator pane, click **Elements**.
2. Select **Search** > **Tag Search**.
3. In the Tag Search window, select the PI Data Archive computers on which you want to search.

   To add or change PI Data Archive computers, choose from the following actions:

- Click ▼ next to the **Server(s)** field and click a PI Data Archive on the list.
- Click ... and double-click on the PI Data Archive you wish to add from the list in the PI Data Archives window.
- To remove a selected PI Data Archive, select the server name in the **Server(s)** field, right-click and click **Delete**.

Servers are added in alphabetical order to the existing selection. You can add multiple selections from either list.

4. You create a PI point search query in the text box next to the **Search** button. A search query comprises a filter name, which can be a text string, a PI point attribute (defaults to **`tag`**) or a PI point value (for example, **`Value`**), an operator (defaults to `:=`), and a query filter (defaults to *, or all PI points). For a full description of the syntax of a PI point query, see Syntax for PI point searches.

   Examples of attribute queries are:

   - `sin*`
   - `tag:=sin*`
   - `tag:<>sin* datatype:float`
   - `step:0 AND pointsource:L`
   - `tag:<>sin* AND PointType:Float64`

   Examples of value queries (which you can combine with attribute queries) are:

   - `Value:>1000`
   - `tag:<>sin* AND Value:>10`
   - `PointType:Int32 AND Value:>10`
   - `PointSource:L AND Annotated:1 AND TimeStamp:t`
   - `creationdate:>y-1d AND future:true AND timestamp:<*`

   By default, PI AF searches for point names that start with the query string (meaning that the **Starts With** filter is already selected).

   Note:
   Queries that contain a query filter name (such as `name:*sine*`), search tag attributes only. In previous versions of PI AF, queries that contained a query filter name would search the descriptor attribute as well as the specified query filter, unless the descriptor attribute was actually specified as part of the query filter.

   To build a search query, choose from the following actions.

To ...	Do this ...
Clear previously selected criteria	Click **Reset**. All queries are cleared, as well as the **Include Description in Search** filter.

Include point descriptions in a query	a. Click . b. Click **Add Criteria** or press ALT+C. c. Select **Include Description in Search** and set the value to **True**.
Type a query directly in the search text box	a. Type the first character of the query. b. Select from the list of matching point attributes and values, or continue typing a point attribute or value manually. c. Enter the criteria to be matched.
Include common PI point attributes in a query	a. Click . b. Enter criteria in any of the following fields: ▪ **Name** (alias for **`tag`** attribute) ▪ **Point Source** ▪ **Data Type** (alias for **`pointtype`** attribute) ▪ **Point Class** (alias for **`ptclassname`** attribute) c. To enter criteria for engineering units and description, click **Add Criteria** or press ALT +C. Then click **Engineering Units** and/or **Description**. Criteria are appended to the string in the search text box, separated by a space. **Note:** The search query returns only PI points that match all criteria.
Include PI point values in a query	a. Click . b. Click **Add Criteria** or press ALT+C. c. Click any of the following criteria: ▪ **Value** ▪ **TimeStamp** ▪ **IsGood** ▪ **Annotated** ▪ **Substituted** ▪ **Questionable** d. Enter criteria in the fields you have selected, as required. Note that you can select a comparison operator for **Value** and **TimeStamp** criteria. Criteria are appended to the string in the search query text box, separated by a space. **Note:** The search query returns only PI points that match all criteria.
Clear an existing query, or remove a criterion from the expanded search area	Click ✕.

Use a previous query	a. Click ▼. b. Click **Recent Searches**. c. Select the existing query from the list.
Filter point names by contents, exact match, or ending	a. Click ▼. b. Click the appropriate filter: ▪ **Contains** ▪ **Exact Match** ▪ **Ends With**

5. Click **Search** to retrieve the points that match into the results table.
6. Choose from the following actions.

To ...	Do this ...
Review time series data	Right-click a PI point and click **Time Series Data**. For more information, see View time series data.
Create a trend	Right-click a PI point and click **Trend**. For more information, see Create trends.
Add a PI point to an existing trend	Right-click a PI point and click **Add to Trend**.
Review current PI point values, time stamps, and attributes	Right-click a PI point and click **Properties**.
Create a PI point data reference	For search results displayed in the palette only. Drag a PI point onto the **Attributes** grid. For more information, see Create direct PI point data references from Tag Search results.
Change the columns in the results table	Click ⚙ and clear the columns that you want to remove.
Add attribute columns to the results table	a. Right-click a column heading and click **Add Column**. b. In the Select Point Attributes window, select the point attributes to be added. c. Click **OK**.
Restore original columns in the results table	Right-click a column heading and click **Restore Original Columns**.

7. Click **OK**.

   The results are displayed in the viewer. Each search is temporarily listed in the browser with a **Tag Search Results** label. To differentiate between searches, you can change a label by right-clicking and clicking **Rename**.

Syntax for PI point searches

Refer to the following sections for details on the syntax for building PI point queries in PI AF and PI Builder. For complete details on PI point query syntax, see "PIPoint Query Syntax Overview" in PI System Explorer **Help** > **AF SDK Reference** > **Overview**, or go to the Tech Support page PIPoint Query Syntax Overview (https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/b8fbb6da-7a4b-4570-a09d-7f2b85ed204d.htm).

Condition filters

To build a PI point query, enter one or more **AND** condition filters that you can also combine with an **OR** condition as needed. Each **AND** condition contains one or more queries, separated by a space or AND. A query consists of a query filter name, an operator, and the query filter. This enables you to specify multiple conditions with a single query, as shown in the following example:

```
(tag:<>sin* AND PointType:Float64) OR (tag:="*Tank*" AND DataType:=Int32)
```

> **Note:**
> You can only use parentheses between **OR** conditions.

You can only reference a filter name once per **AND** condition of the query string. For example, `PointId:>5 AND PointId:<10` generates an error, whereas `PointType:=Int32 OR PointType:=Float32` is valid.

For maximum efficiency, build your query so that you eliminate most items from the retrieved results with your first condition filters.

Query filter names

When querying based on PI point attributes, the query filter name is a PI point attribute name or alias. Common aliases are:

Alias name	Attribute name
Name	Tag
DataType	PointType
Description	Descriptor
PointClass	PtClassName

Starting in PI AF 2017, you can query based on values, in addition to querying PI points based on attribute. However, you cannot use the **OR** condition to query a PI point value. For example, you would generate an error if you were to enter the following queries:

- `IsGood:false OR Annotated:true`
- `PointType:Float AND Value:>10` because `PointType:Float` is implicitly translated to `'PointType:=Float16 OR PointType:=Float32 OR PointType:=Float64'`
- `PointType:Int AND Value:>10` because `PointType:Int` is implicitly translated to `'PointType:=Int16 OR PointType:=Int32'`
- `sin* AND Value:>10` because `sin*` is implicitly translated to `'tag:=sin* OR Descriptor:=sin*'` if the default filter setting for **Include Description in Search** is selected. To be valid, you would need to clear the **Include Description in Search** filter.

Wildcard characters

You can use the following special characters in a PI point query.

> **Note:**
> Results of the examples below assume you are using the default search option, which searches for PI point names that start with your search string.

Special character	Description	Example
*	Substitute any number of unspecified characters	`sin*` Returns all PI points that have names starting with "sin", for example, `sinusoid` and `sinusoidu`.
?	Substitute a single unspecified character	`CD?158` Returns all PI points that have names starting with "CD", followed by any single character, followed by "158" (for example, `CD1158`, `CDA158`, and so on).

`:` or `:=`	When searching for all PI points with a specific attribute value (other than name), separates the attribute and the value you are searching for.
	Tip: When searching for a PI point name that contains a colon, enclose the name in double quotation marks, or precede the colon with a backslash.
		`pointsource:R`
		Returns all PI points that have the **`pointsource`** value `R`.
		`"ba:temp.1"`
		`ba\:temp.1`
		Either of the above examples returns the PI point named `ba:temp.1`.

`''`
or
`""`
	Delimiters for search strings containing spaces or special characters	`'*Owner Change*'`
		or
		`"*Owner Change*"`
		Returns all PI points that have names containing Owner Change.
		`"ba:temp.?"`
		Returns all PI points that have names starting with `ba:temp.` and ending with any single character.

Operators

The following table lists the operators that you can use in an **AND** condition.

Operator	Description	Example
`=`	The **`EQUALS`** operator.	`Tag:Tank*` or `Tag:=Tank*`
`<>`	The **`NOT EQUALS`** operator.	`PointType:<>Int32`
`<`	The **`LESS THAN`** operator.	`Descriptor:<M`
`<=`	The **`LESS THAN OR EQUAL`** operator.	`Tag:<=Tank`
`>`	The **`GREATER THAN`** operator.	`Tag:>Tank`
`>=`	The **`GREATER THAN OR EQUAL`** operator.	`Tag:>=Tank`

In PI point value queries with a String data type, you cannot use the following operators: `<`, `<=`, `>`, or `>=`. Furthermore, when boolean values are expected (as with **`Substituted`**,

`Questionable`, **`Annotated`**, and **`IsGood`** point value queries), you can only use the = and <> operators.

Syntax restrictions

- You cannot query future point attributes, such as `creationdate:>y-1d AND future:true`, on PI Data Archive servers older than 3.4.395.
- You cannot query security point attributes, such as **`PtSecurity`** and **`DataSecurity`**, on PI Data Archive servers older than 3.4.380.

Manage search results

PI System Explorer displays all search results in the browser where you can save them in a searches collection for reuse, rename them, or edit the criteria. All unsaved search results are indicated by an asterisk.

Note:
All unsaved search results are deleted when you exit PI System Explorer.

Procedure

1. Optional. To work from a list of search results in the viewer rather than in the browser, choose from the following actions:

To ...	Do this ...
Manage an element or attribute search	a. In the navigator, click **Elements**. b. In the browser, click the **Element Searches** collection.
Manage an event frame search	a. In the navigator, click **Event Frames**. b. In the browser, click the **Event Frame Searches** collection.
Manage a transfer search	a. In the navigator, click **Event Frames**. b. In the browser, click the **Transfer Searches** collection.

2. Select a search and choose from the following actions:

To ...	Do this ...
Save a search result	Right-click an unsaved search result name and click **Save**.
Rename a search result	a. Right-click a search result name and click **Rename**. b. Type a new search result name.
Copy a search result	Right-click the search result name and click **Copy**.
Edit criteria for a search result	a. Right-click a search result name and click **Properties**. (In the viewer, you can also double-click a search result name.) b. Modify search criteria as needed and click **OK**.

Delete a search result	a. Right-click a search result name and click **Delete**. b. Click **Yes** in the Delete confirmation window.
Rearrange event frame or transfer search results	In the browser only: a. Right-click an event frame or transfer search result name and click **Arrange By**. b. Select from the following options: ▪ **Arrange By Name** ▪ **Arrange By End Time** ▪ **Arrange By Start Time** ▪ **Arrange By Category** ▪ **Arrange By Template**
Display the full path to elements or event frames	In the viewer only: a. Right-click on the column header (or below the search results grid). b. Click **Show Full Paths**.

Search for elements

Use the Element Search window to retrieve element data that matches your search criteria.

Procedure

1. In the navigator pane, click **Elements**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Element Search**.
   - In the browser, right-click the **Element Searches** root node and select **New Search**.
3. Set the Element Search window to find the desired element or elements in the PI AF database.

   Note:
   If you enter values for multiple criteria, the search returns only those elements that match all the specified criteria.

4. From the **Actions** ▾ list, select the type of filter to apply: **Contains**, **Exact Match**, **Starts With**, or **Ends With**.
5. Choose from the following actions:

To ...	Do this ...
Type one or more filter conditions directly into the **Enter element criteria** field	◦ Use the : or := operator to select elements that match a name, description, template, and category. ◦ Use special characters as needed. For more information, see Special characters in name searches. ◦ Enclose strings that contain spaces or special characters with the " character. ◦ Separate filter conditions with a whitespace, for example: `Category:"Processing Plants" Name:Cracking`

Filter elements under **Criteria**	Enter values in the following fields as needed. Click **Add Criteria** to enter criteria for additional fields.

- **Name**

  Enter the name of the element to retrieve, based on the filter type. You can enter special characters to match part of a name. See Special characters in name searches.

- **Element Search Root**

  Enter the element that you want to use as the root or base level for the element search. Type the exact name or click ... to open the Element Browser window, where you can view the element hierarchy and select an element. You cannot include wildcard characters in the entered name. If you do not specify an element, you set the main level of the element hierarchy as the root.

  Depending on your PI AF hierarchy, specifying an element in the **Element Search Root** field can improve search performance.

- **All Descendants**

  Select **True** to retrieve any sub-element in the hierarchy that matches the specified criteria. Select **False** to retrieve only root-level elements that match the specified criteria.

- **Template**

  Select the template that retrieved elements must be based on. After you select a template, you can add criteria to find elements by attribute value.

- **Category**

  Select the category that retrieved elements must match.

- **Attribute Value**

  You can specify up to three conditions for an attribute value. For each condition, specify an attribute name, operator, and attribute value, such as `Temperature >= 98`.

  - In versions prior to PI AF 2018, only available when you specify a template. For more details, see Configure search conditions for attribute values when a template is specified.
  - In PI AF 2018 and later versions, you can specify attribute values without needing to specify a template. For more details, see Configure search conditions for attribute values when no template is specified.

- **Description**

  Enter a string (of up to 440 characters) that retrieved elements must match.

- **Element Type**

  Select the type that retrieved elements must match.

- **Is Annotated**

  (PI AF 2017 or later versions.) Set to `True` to retrieve only elements that have annotations. Set to `False` to retrieve only elements that do not have annotations.

- **Creation Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve elements that were created in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Creation Date** a second time and filter the search for results between two values (`< *+1d` is the default).

- **Modify Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve elements that were modified in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Modify Date** a second time and filter the search for results between two values (`< *+1d` is the default).

> **Note:**
> An element's modify date is updated whenever an annotation or child element is added, as well as when a change to its configuration is checked into the database.
>
> Most template changes, or any modification to an attribute value that is not a configuration item, will not affect an element's modify date.

- **Results per Page**

  Enter the maximum number of elements to show on a single page of the search results.

6. Optional. Specify how you want results to be displayed in the **Results** table.

To ...	Do this ...
Group by template	Select the **Template** check box.
Group by category	Select the **Category** check box.
Change column selections	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Column Visibility**. c. Select or clear column selections as needed.
Display attributes as columns	a. Click and click **Select Attributes**. b. In the Select Attributes window, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) attributes. c. Click . d. Click **OK**.
Display full paths of elements	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Show Full Paths**.
Conceal full paths of elements	a. Right-click the column heading in the **Results** table or the white space below. b. Click **Hide Full Paths**.

7. Select any of the results you want to use and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Topics in this section

- Configure search conditions for attribute values when a template is specified
- Configure search conditions for attribute values when no template is specified

Configure search conditions for attribute values when a template is specified

You can restrict your search based on the value of an attribute. After you specify a template (required in versions prior to PI AF 2018), use the **Attribute Value** field to configure up to three conditions that the search must match regarding an attribute value.

Before you start

- Unindexed attributes can take a significant amount of time to evaluate, particularly if they are configured with a data reference.
- You cannot search for attributes with `Object` or `Array` value types.

Procedure

1. Click ... (**Display attribute choices**) and select an attribute from the list of possible attribute categories:

Option	Description
attribute a	Indexed attributes, including configuration attributes.
Configuration Attributes	Configuration attributes that are not indexed.
Other Attributes	Non-configuration attributes that are not indexed.

2. Click >=, and select a mathematical operator from the list.
   - For attribute value types `Single` and `Double`, queries do not support the `In` operator.
   - For attribute value types `String`, `Boolean`, and `Int64`, queries do not support the following operators:
     - < (less than)
     - > (greater than)
     - <= (less than or equal to)
     - >= (greater than or equal to)
3. Enter a value in the units specified by the default UOM in the attribute template.

> **Note:**
> For indexed attributes that store `String` value types, the search only uses the first 40 characters of the entered value.

Configure search conditions for attribute values when no template is specified

When using a PI AF 2018 or later client and connected to a PI AF 2018 or later server, you can restrict your search based on the value of an attribute, without first having to specify a template. Use the **Attribute Value** field to configure up to three conditions that the search must match regarding an attribute value.

Before you start

- When no template is selected, all attributes are searched, including those used in templates.
- When no template is selected, wildcard characters should not be used in the **Attribute Value** field.
- Unindexed attributes can take a significant amount of time to evaluate, particularly if they are configured with a data reference.
- You cannot search for attributes with `Object` or `Array` value types.

Procedure

1. From the **Add Criteria** list, select **Attribute Value**.
2. In **Attribute Value**, enter the name of the attribute to retrieve.
3. Click >=, and select a mathematical operator from the list.
4. Enter a value. If the value type is `Enumeration Set` or `Guid`, you also need to append `as` *`Enumeration_Set`* or `as Guid` to the search query displayed in the **Enter element criteria** field. For example, to search for assets with a Health Status attribute value of Error, you would enter the following as the **Attribute Value** criteria:

   To complete the search query, you would append the name of the Health Status enumeration set to the string already displayed in the **Enter element criteria** field:

   ```
   as 'Health Status'
   ```

   The completed search query in the **Enter element criteria** field would look as follows:

   Note:
   You also need to append `As String` or `As Numeric` whenever the value type of an attribute does not match what the query search value appears to be. For example, if the value type of an attribute were `String` and the search value were `55` (which looks like a numeric), you would need to append `As String` to the query.

   Note:
   You can use uppercase and lowercase combinations when you append to search queries: `As String`, `as string`, and `as STRING` are all equivalents. You can also use either `'` or `"` to enclose enumeration sets and strings that contain space characters.

Special characters in name searches

When searching for objects by name, such as element names, event frame names, or attribute names (when associated with a template), you can use special characters:

Special character	Purpose
*	Substitute any number of unspecified characters.
?	Substitute a single unspecified character.
[*xyz*]	Specify a set of characters (*x*, *y*, or *z*) to match.
[!*xyz*]	Specify a set of characters (*x*, *y*, or *z*) to preclude a match.
\	Ignore the subsequent special character and interpret as its actual character.
[*first-last*]	Specify a range of characters (from *first* to *last*) to match. For example, `a[a-c]` matches `aa`, `ab`, or `ac`, but does not match `ad` or `abc`.

Search for attributes on elements

You can search for an attribute or group of attributes. You may want to locate a particular attribute, for example, to configure a formula data reference or to assign it as a value to another attribute. (To search for a specific attribute value, choose an attribute value as search criteria for an element search.)

Procedure

1. In the navigator pane, click **Elements**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Attribute Search**.
   - In the browser, right-click the **Element Searches** collection and select **New Attribute Search**.
3. Set the Attribute Search window to find the desired attributes in the PI AF database:
   a. Under **Where**, set the fields to restrict attributes retrieved:
      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.
      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.
      - **Attribute category**

        Select the category that retrieved attributes must match.
      - **Attribute value type**

        Select the type of value that the retrieved attributes must store.

- **Maximum results**

  Enter the maximum number of matching attributes to retrieve.

b. In the **Element Criteria** area, set the fields to restrict the elements searched for matching attributes:

- **Search Root**

  Enter the element that you want to use as the root or base level for the attribute search. Type the exact name or click **Browse** ... to open the Element Browser window, where you can view the element hierarchy and select an element. You cannot include wildcard characters in the entered name. If you do not specify an element, you set the main level of the element hierarchy as the root. Depending on your PI AF hierarchy, specifying an element in the **Search Root** field can improve search performance.

  Select the **Search Sub-Elements** check box to search the entered root and any sub-elements. Clear this check box to search only the entered root.

- **Name**

  Enter the name of the elements in which you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

- **Description**

  Enter a string (of up to 440 characters) to retrieve only those elements that match.

- **Category**

  Select the category of the elements in which you want to search for attributes.

- **Template**

  Select the template of the elements in which you want to search for attributes.

- **Type**

  Select the type of the elements in which you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table.

   Alternatively, use the element tree under **Search results** to browse for attributes under particular elements, and then select the attributes to add them to the **Search results** table.

   Remember that the attributes available by searching and the attributes available by browsing the element hierarchy depend on the configuration properties of the attributes.

5. Select items in the search results list and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for event frames

Use the Event Frame Search window to retrieve event frame data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Event Frame Search**.
   - In the browser, right-click the **Event Frame Searches** collection and select **New Search**.
3. Set the Event Frame Search window to find the desired event frame or event frames in the PI AF database.

   Note:
   If you specify values for multiple criteria, the search returns only those event frames that match *all* the specified criteria.

4. From the **Actions** ▾ list, select the type of filter to apply: **Contains**, **Exact Match**, **Starts With**, or **Ends With**.
5. Choose from the following actions:

To ...	Do this ...
Type one or more filter conditions directly into the **Enter Event Frame Criteria** field	◦ Use the : or := operator to select elements that match a name, description, template, primary element name, or category. ◦ Use special characters as needed. For more information, see Special characters in name searches. ◦ Enclose strings that contain spaces or special characters with the " character. ◦ Separate filter conditions with a whitespace, for example: `Category:"Processing Plants" Name:Cracking`
Filter event frames under **Criteria**	Enter values in the following fields as needed. Click **Add Criteria** to enter criteria for additional fields.

   - **Search**

     Select the method you want to use to specify when the desired event frames occurred. The window shows the appropriate fields for the selected method.

     For example, select **Active Between** to specify a start time and end time, and find event frames active any time during that period. Select **Starting After** to specify only a start time, and find event frames that start after that time.

   - **In Progress**

     If available, select this check box to further restrict matching event frames to those that have not yet finished.

- **Search start**

  A PI time expression that specifies the start of the time period used to search for event frames.

- **Search end**

  A PI time expression that specifies the end of the time period used to search for event frames.

  Optional. From the list next to this field, select a defined time to automatically fill in the **Search start** and **Search end** fields.

- **All Descendants**

  Select this check box to search all levels of the event frame hierarchy below the specified root for matching event frames.

- **Name**

  Enter the name of the event frame to retrieve, based on the filter type. You can enter special characters to match part of a name. See Special characters in name searches.

- **Element Name**

  Enter a PI AF element that must be the parent of any retrieved event frames. You can enter special characters to match part of a name. See Special characters in name searches.

- **Category**

  Select the category that retrieved event frames must match.

- **Results per Page**

  Enter the maximum number of event frames to show on a single page of the search results.

- **Template**

  Select the template that retrieved event frames must be based on. After you select a template, you can add criteria to find elements by attribute value.

- **Analysis Name**

  Enter the name of the analysis that retrieved event frames were generated from. You can use wildcards as needed.

- **Attribute Value**

  You can specify up to three conditions for an attribute value. For each condition, specify an attribute name, operator, and attribute value, such as `Temperature >= 98`.

  - In versions prior to PI AF 2018, only available when you specify a template. For more details, see Configure search conditions for attribute values when a template is specified.
  - In PI AF 2018 and later versions, you can specify attribute values without needing to specify a template. For more details, see Configure search conditions for attribute values when no template is specified.

- **Duration**

  Select an operator and enter a value, which can include a PI time abbreviation. For example, select **>=** and enter `1d` to retrieve events that last at least one day. You can select **Duration** a second time and filter the search for results between two values. For example, select **<=** and enter `2d` to retrieve events that lasted between one and two days.

- **Event Frame Search Root**

  Enter the event frame that you want to use as the root or base level for the event frame search. Type the exact name or click **Browse** ... to open the Event Frame Browser window, where you can view the event frame hierarchy and select an event frame. You cannot include wildcard characters in the entered name. If you do not specify an event frame, you set the main level of the event frame hierarchy as the root. Depending on the complexity of your PI AF hierarchy, specifying an event frame in the **Event Frame Search Root** field can improve search performance.

- **Can Be Acknowledged**

  (PI AF 2016 or later versions.) Set to `True` to retrieve event frames that can be acknowledged. The ability to acknowledge event frames is determined in the template on which the event frame is based.

- **Is Acknowledged**

  (PI AF 2016 or later versions.) Set to `True` to retrieve event frames that have been acknowledged. Set to `False` to retrieve only event frames that have not been acknowledged.

- **Is Annotated**

  (PI AF 2016 or later versions.) Set to `True` to retrieve only event frames that have annotations. Set to `False` to retrieve only event frames that do not have annotations.

- **Severity**

  (PI AF 2016 or later versions.) Select an operator and select a severity setting from the list. For example, select **>= Minor** to retrieve event frames that have at least a Minor severity setting. You can select **Severity** a second time and filter the search for results between two severity settings. For example, select **<= Critical** to retrieve event frames that have Minor, Major and Critical severity settings.

- **Creation Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve event frames that were created in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Creation Date** a second time and filter the search for results between two values (`< *+1d` is the default).

- **Modify Date**

  (PI AF 2017 or later versions.) Select an operator and enter a date or a PI time abbreviation (`>= *-30d` is the default) to retrieve event frames that were modified in the specified period. You can also click to select a date in the Date and Time Picker window. You can select **Modify Date** a second time and filter the search for results between two values (`< *+1d` is the default).

> **Note:**
> An event frame's modify date is updated whenever a capture value, an annotation, or a child event frame is added, as well as when a change to its configuration is checked into the database.
>
> Most template changes, or any modification to an attribute value that is not a configuration item, will not affect an event frame's modify date.

6. Optional. Specify how you want results to be displayed in the **Results** table.

To ...	Do this ...
Group by template	Select the **Template** check box.
Group by category	Select the **Category** check box.
Change column selections	a. Right-click the column heading in the **Results** table or white space below. b. Click **Column Visibility**. c. Select or clear column selections as needed.
Display attributes as columns	a. Click and click **Select Attributes**. b. In the Select Attributes window, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) attributes. c. Click . d. Click **OK**.
Display full paths of event frames	a. Right-click the column heading in the **Results** table or white space below. b. Click **Show Full Paths**.
Conceal full paths of event frames	a. Right-click the column heading in the **Results** table or white space below. b. Click **Hide Full Paths**.

7. Click **Search** to retrieve the matching event frames into the **Results** table.
8. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for attributes on event frames

Use the Event Frame Attribute Search window to retrieve event frame attribute data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Event Frame Attribute Search**.
   - In the browser, right-click the **Event Frame Searches** collection and select **New Attribute Search**.

3. Set the Event Frame Attribute Search window to find the desired attributes in the PI AF database:

   a. Under **Where**, set the fields to restrict attributes retrieved:

      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.

      - **Attribute category**

        Select the category that retrieved attributes must match.

      - **Attribute value type**

        Select the type of value that the retrieved attributes must store.

      - **Maximum results**

        Enter the maximum number of matching attributes to retrieve.

   b. In the **Event Frame Criteria** area, set the fields to restrict the search to matching attributes:

      - **Search**

        Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

      - **Search start / Search end**

        Choose or enter start and end times for the search time period.

      - **Search Root**

        Enter the event frame to use as the root or base level for the attribute search. Type the exact name or click **Browse** ... to open the Event Frame Browser window, where you can view the hierarchy and select an event frame. You cannot include wildcard characters in the entered name. If you do not specify an event frame, the root is set to the main level of the hierarchy. Depending on the complexity of your PI AF hierarchy, specifying search root can improve search performance.

        Select the **Search Sub-Event Frames** check box to search the entered root and its children. Clear this check box to search only the entered root.

      - **Name**

        Enter the name of the event frames you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Description**

        Enter a description string of up to 440 characters to retrieve only those elements that match.

- **Duration**

  Enter minimum and maximum duration values to limit the attribute search to event frames whose duration is within these limits.

- **Category**

  Select the category of the event frames you want to search for attributes.

- **Template**

  Select the template of the event frames you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table. You can limit the results further to list only those attributes that match the text entered into the **Search results** field.
5. Select items in the search results list and click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for transfers

Use the Transfer Search Criteria window to retrieve transfer data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Transfer Search**.
   - In the browser, right-click the **Transfer Searches** collection and select **New Search**.
3. Set the Transfer Search Criteria window to find the desired transfers in the PI AF database:
   - **Search**

     Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

   - **Search start / Search end**

     Choose or enter start and end times for the search time period.

   - **Name**

     Enter the name of the transfers you want to search for. You can enter special characters to match part of a name. See Special characters in name searches.

   - **Description**

     Enter a description string of up to 440 characters to retrieve only those transfers that match.

- **Source**

  Enter or click ... to open the Element Browser window, where you can view the element hierarchy and select the source element for transfers you want to include in the search. Click the **Any** button to reset to the default <Any>.

- **Destination**

  Enter or click ... to open the Element Browser window, where you can view the element hierarchy and select the destination element for transfers you want to include in the search. Click the **Any** button to reset to the default <Any>.

- **Template**

  Select the template of the transfers you want to search for.

- **Maximum results**

  Enter the maximum number of matching transfers to retrieve.

If you specify values for multiple settings, the search returns only those transfers that match all the specified settings.

4. Click **Search** to retrieve the matching transfers into the **Search results** table.

   You can limit the results further to list only those transfers that match the text entered into the **Search results** field.

5. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Search for attributes on transfers

Use the Transfer Attribute Search window to retrieve transfer attribute data that matches your search criteria.

Procedure

1. In the navigator pane, click **Event Frames**.
2. Choose from the following actions:
   - From the PI System Explorer menu, select **Search** > **Transfer Attribute Search**.
   - In the browser, right-click the **Transfer Searches** collection and select **New Attribute Search**.
3. Set the Transfer Attribute Search window to find the desired attributes in the PI AF database:
   a. Under **Where**, set the fields to restrict attributes retrieved:
      - **Attribute name**

        Enter the name of the attribute to retrieve. You can enter special characters to match part of a name. See Special characters in name searches.

      - **Attribute description**

        Enter a string (of up to 440 characters) that retrieved attributes must match.

- **Attribute category**

  Select the category that retrieved attributes must match.

- **Attribute value type**

  Select the type of value that the retrieved attributes must store.

- **Maximum results**

  Enter the maximum number of matching attributes to retrieve.

b. In the **Transfer Criteria** area, set the fields to restrict the attribute search to transfers matching the specified criteria:

- **Search**

  Choose criteria to find transfers that overlap the time period specified by **Search start** and **Search end**.

- **Search start / Search end**

  Choose or enter start and end times for the search time period.

- **Name**

  Enter the name of the transfers you want to search for attributes. You can enter special characters to match part of a name. See Special characters in name searches.

- **Description**

  Enter a description string of up to 440 characters to retrieve only those attributes that match.

- **Source**

  Enter or browse to and select the source element for transfers whose attributes you want to include in the search.

- **Destination**

  Enter or browse to and select the destination element for transfers whose attributes you want to include in the search.

- **Template**

  Select the template of the transfers you want to search for attributes.

- **Category**

  Select the category of the transfers you want to search for attributes.

If you specify values for multiple settings, the search returns only those attributes that match all the specified settings.

4. Click **Search** to retrieve the matching attributes into the **Search results** table.

   You can limit the results further to list only those attributes that match the text entered into the **Search results** field.

5. Click **OK**.

   The results are displayed in the viewer and in the browser tree. For more information on working with search results, see Manage search results.

Trends in PI System Explorer

While browsing elements in PI System Explorer, you can quickly create simple trends to verify the formulas, calculations, and other measurements you are building within PI AF. For example, while you are designing a PI AF formula, you might want to see what the results look like over the last several hours. Or, if you are creating PI point attributes that involve summaries, it could be useful to see what those look like over recent hours.

You can add the PI AF attributes or PI points of interest to a trend; one way of adding them is to drag and drop them onto the trend. You can then explore the data by adjusting the duration and the start and stop times of the trend.

After you have created the trend, it remains available in the Trend window even when you are using other features within PI System Explorer. You can add or remove traces to the trend at any point. You can also right-click the trend cursor and select commands from a context menu.

You can also see a trend of an attribute when you view its time series data (by right-clicking an attribute and clicking **Time Series Data**). The trend is automatically shown at the bottom of the Time Series Data window.

Create trends

To verify formulas, calculations, and other measurements in PI AF, create a trend.

Before you start

OSIsoft recommends you use visualization tools, such as PI Vision, PI ProcessBook, and PI WebParts, to create and save trends that you plan to use more than once.

Procedure

1. From the PI System Explorer menu, select **View** > **Show Trend**.
2. In the Trend window, choose from the following actions.

To ...	Do this ...
Add an attribute to the trend	a. Click **Add Attributes**. b. In the Attribute Search window, specify search parameters in the **Attribute Name**, **Attribute Category**, and other fields to find the attribute of interest. c. Expand the element hierarchy in the Search results area and select the attribute. d. Click **OK.** The Trend window displays a trend showing how that attribute has varied over the past hour (1 hour is the default duration).
Adjust the duration	Specify times in the **Start Time** and **End Time** fields.

Add a PI point to the trend	a. Click **Add PI Points**. b. In the Tag Search window, search for the PI point you want to see and select it in the results. c. Click **OK**. The PI point is added to the trend.
Adjust which traces are visible on the trend	Click **Traces** and clear the check box of attributes or points you do not want to see.

3. Add any further PI AF attributes and PI points for which you want to view data.

   The traces in the Trend window persist until you close the window.

   You can drag attributes and PI points from other search results and drop them on the trend to add them. Alternatively, you can right-click an attribute or point and click **Add to Trend**.

Version management for equipment and processes

When you need to make an equipment change, such as swapping out a broken pump for a similar pump, you can continue to use the existing element to represent the new pump. However, to indicate that a change has occurred, it is a good idea to create a new version of the element. That way you have a record of the change to the equipment that the element represents.

Similarly, you might need to create versions of a table. For example, strapping tables of tanks are updated periodically to match the changing physical characteristics of the tank. By versioning the table, you will be able to use the table as it was configured at the time of the data collection.

PI System Explorer allows you to create multiple versions of elements, models, and tables.

You can also indicate an element, model, or table is not in service past a specific date by selecting that date as its *obsolete* date.

Videos

For an overview of version management, watch this video:

https://www.youtube.com/watch?v=8cHFHW1SID0&rel=0

For information on how to create versions of elements and tables, watch this video:

https://www.youtube.com/watch?v=j2ilXV2XmV0&rel=0

Topics in this section

- Create versions for elements or models
- Create table versions
- Compare two versions

Create versions for elements or models

Create a new version when you need to keep track of changes to an asset.

Before you start

Whenever a new version is created, a check-in occurs on the element or model.

Procedure

1. In the navigator, click **Elements**.
2. in the Elements browser, choose from the following actions.
   - Right-click the element or model you want to version and click **Create Version**.
   - Click the element or model you want to version. In the viewer, select the **Version** tab and click **New Version**.
3. In the Create Version window, enter an **Effective Date** for the version or click to select a date. PI System Explorer creates a new version.
4. Optional. In the **Comment** field, enter information on the new version.
5. Click **OK**.

   To indicate that more than one version exists, the icon in the browser changes to for an element or for a model.

Create table versions

Create a new version when you need to keep track of changes to a table.

Before you start

Whenever a new version is created, a check-in occurs on the table.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, choose from the following actions.
   - Right-click the table you want to version and click **Create Version**.
   - Click the table you want to version. In the viewer, select the **Version** tab and click **New Version**.
3. In the Create Version window, enter an **Effective Date** for the version or click to select a date. PI System Explorer creates a new version.
4. Optional. In the **Comment** field, enter information on the new version.
5. Click **OK**.

   To indicate that more than one version of the table exists, the icon in the browser changes to .

Compare two versions

You can compare two versions of an element, model, or table that has more than one version.

Procedure

1. Choose from the following actions.

To...	Do this...
Compare table versions	a. In the navigator, click **Library**. b. In the Library browser, select a table that displays the icon.
Compare element or model versions	a. In the navigator, click **Elements**. b. In the Elements browser, select an element that displays the icon, or a model that displays the icon.

2. In the viewer, select the **Version** tab and click **Show History**.
3. In the Show History window, select two versions in the left panel from the list of all existing versions.

   The right panel uses colored change bars to indicate the differences between the versions:

   - Red means that the item was present in the older version and is not present in the newer version.
   - Green means the item is present in the newer version but was not present in the older version.
   - Yellow means that the item changed between the older and newer version.
4. To exit, click **Close**.

Display of different object versions and obsolete objects

By default, PI System Explorer shows the latest versions of elements, models, and tables (even if those versions are in the future). It also excludes earlier versions and all objects that have an **Obsolete** date specified (even if that date is in the future). As a result, when you specify an **Obsolete** date for an object and check it in, it immediately disappears from PI System Explorer.

You can, however, find and view an obsolete object, or a particular version of an object, by setting **Query Date** to a fixed time at which the object or version is current. To display an element that became effective January 1, 2010 and obsolete on January 1, 2011, for example, set **Query Date** to any time between those two dates.

Video

For information on how to display different versions, watch this video:

https://www.youtube.com/watch?v=MwFrhzjPrns&rel=0

Topics in this section

- Query Date and PI System Explorer time
- Show specific dates and times
- Time context for attribute values

Query Date and PI System Explorer time

Query Date determines the time values PI System Explorer uses to:

- Find and display attribute values.
- Identify current versions of objects to display and include in search results.

At the default **Query Date** setting, **Set to Latest**, PI System Explorer finds and displays the most current data for attribute values. You can override the **Query Date** to view values for a different time with **Set Time Context**. See Set time context for displayed attribute values.

Set to Latest also causes PI System Explorer to use the latest versions of objects, even if those versions are in the future, and excludes any object with an **Obsolete** date, even if that date is in the future.

You can change **Query Date** to specify a fixed date and time:

- The date and time picker lets you choose a specific date and time setting.
- The selections **Set to Now** and **Set to Today** each use the current time to establish a fixed date and time. For example, if you choose **Set to Now** on Monday, then on Wednesday **Query Date** will still be set to Monday.

How Query Date affects PI System Explorer

What is affected	Query Date is *Set to Latest*	Query Date is a fixed date and time
All PI System Explorer objects	PI System Explorer uses the latest versions, even if those versions are in the future; it excludes any object with an obsolete date, even if that date is in the future.	PI System Explorer uses versions of objects that are in effect at that time.
Element, model, and table versions in search	PI System Explorer returns the latest versions, even if those versions are in the future.	PI System Explorer returns the versions that are in effect at that time.
Relative times in searches	PI System Explorer defines the current time (*) as the time of the search.	PI System Explorer defines the current time (*) as the time specified by the Query Date.
PI data	PI data is returned at its latest snapshot value.	PI data returned is the value for the timestamp specified by the Query Date.

Show specific dates and times

When you set the query date to a specific date and time, it remains constant at that date and time until you change it again.

Procedure

1. On the PI System Explorer tool bar, click the **Query Date** button.
2. In the Date and Time Picker window, choose the date and time that you want. If you want to set the query date to the current date and time, click the **Set to Now** button.

> **Note:**
> When you choose **Set to Now** as the query date, you are setting the query date to a specific date and time; this is a constant value. This is not the same as setting the query date to always be the current time. For that, you should choose the **Set to Latest** option.

3. Click **OK**.

Video

For information on how to set a specific query date, watch this video:

https://www.youtube.com/watch?v=_381ZTzkFo4&rel=0

Time context for attribute values

The time or time range that PI System Explorer uses to display attribute values is determined by values you set in the Set Time Context window.

By default, PI System Explorer displays the latest values for attributes based on the current time. The Query Date reflects PI System Explorer's time setting (see Query Date and PI System Explorer time). You can view attribute values for a different time, however, by specifying a new time in the Set Time Context window.

The time and time range settings in the Set Time Context window apply only to displayed attribute values. Query Date continues to apply to all other PI System Explorer objects.

Set time context for displayed attribute values

You use Set Time Context to specify a different time or time range for displayed attribute values.

Procedure

1. On the PI System Explorer toolbar, click .

   Alternatively, click **Tools** > **Options** and select the **Time Context** tab.

2. In the Set Time Context window, select one of the following options. Enter time values for an option directly, or click to select a date and time. You can also click to construct a time expression, such as `*-1d`.

Option	Description
Query Date Time	Displays the current setting for Query Date (defaults to `Latest Available`). You can specify a different date time setting for Query Date, to enable PI System Explorer to evaluate which versions of objects to retrieve from the PI AF database.
Alternative Time	Displays the time context for data retrieval after applying the current Query Date setting. The default is the current time (*), but you can enter a new time context to use for displayed attribute values.

Time Range	Enter Start and End times to specify a new time range to use for displayed attributed values that require a time range.

3. Optional. Click **Set as Default** to register the current settings as the default time context, or click **Restore Default** to restore the time context for **Query Date Time** to `Latest Available`.
4. Click **Apply** to see your changes and **OK** to keep them. The revised time context settings are displayed in the PI System Explorer title bar.

View time series data

PI System Explorer can display time series data for attributes that contain or reference such data. This allows you to preview how your data reference configurations will work with your collected data in other PI client tools, such as PI ProcessBook, PI Vision, PI DataLink and PI WebParts.

Note:
Some attribute configurations do not support this type of data. See Restrictions on viewing time series data.

Procedure

1. Right-click on the attribute in the viewer and click **Time Series Data**.
2. In the Time Series Data window, choose the tab representing the type of data you want to see:
   - **Archive**

     For PI points. Shows the archived values within the specified time. Settings include Boundary type and Filter expression.
   - **Sampled**

     Returns evenly-spaced, interpolated values over a regular interval. You can include a Filter expression.
   - **Plot**

     Retrieves values over the specified time range suitable for plotting over the specified number of intervals. Typically, the intervals represent pixels and you would use this feature to represent the screen width available for the plot.
   - **Summary**

     Displays summary Statistics for attribute values that support this feature. You can specify Weighting.
   - **Data Pipe**

     Enables you to monitor and display data changes within attributes. For attributes that support future data, you can specify how the data pipe returns data with the Event horizon mode and Offset properties. For historical tags, Event Horizon Mode has no effect.

Topics in this section

- Boundary type

- Filter expression
- Statistics
- Weighting
- Event horizon mode
- Restrictions on viewing time series data

Boundary type

Specify a boundary type to determine how the searches for data values are handled near the start and end times of the value range:

- **Inside (default)**

  Returns values at start and end times, if they exist, or the nearest values occurring within the range.

- **Outside**

  Returns the closest values occurring immediately outside the range.

- **Interpolated**

  Returns interpolated values at start and end times.

Filter expression

Add a filter expression to filter event values using a mathematical expression, eliminating data for which the expression evaluates as false. For example, the simple filter expression:

```
'.' < 70
```

would remove all values over 70 from the calculation. You can also use any valid PI performance equation in the filter expression to build more complex expressions to remove atypical peaks in data values, for example.

Expression variables are limited to attributes or PI points which originate from a single PI Data Archive. Attributes which resolve to a static value (no data reference configured), are also acceptable. See Indirect PI point references for a complete description of possible reference syntax.

Attribute variable examples

Filter expression	What it does
`'.'`	Reference to the attribute being queried.
`'Level'`	Reference to the attribute `Level` at the same attribute hierarchy level in the element or event frame.
`'..'`	Reference to the parent attribute of the attribute being queried. Only valid for nested child attributes.
`'.|HighLimit'`	Reference to the child attribute `HighLimit` of the attribute being queried.
`'|Temperature'`	Reference to the `Temperature` attribute at the top hierarchy level in the current element or event frame.
`'\\MyPIServer\sinusoid'`	Absolute path to a PI point. It must be in the same PI Data Archiveas the current queried attribute.

Filter expression	What it does
`'\\myAFServer\myDatabase\myElement|myAttribute'`	Absolute path to a PI AF attribute.
`'\myRootElement|myAttribute'`	Database relative path to a PI AF attribute.

PI point variable examples

Filter expression	What it does
`'.'`	Reference to the PI point being queried.
`'sinusoid'`	Reference to the PI point `sinusoid` in the same PI Data Archive.

Statistics

The **Summary** tab provides the following statistics for attributes that support such statistics:

- **Percent Good**

  Displays the percentage of time for which good values are returned over the total time range. Good values are event values determined to be valid and not in an error state.

- **Average**

  Computes the average of values during the interval.

- **Minimum**

  Returns the minimum value during the interval.

- **Maximum**

  Returns the maximum value during the interval.

- **Range**

  Computes the maximum value minus the minimum value during the interval.

- **Standard deviation**

  Computes the standard deviation of values during the interval.

- **Population standard deviation**

  Computes the population standard deviation of values during the interval.

- **Count**

  Returns the number of values stored during the interval.

Weighting

The **Summary** tab allows you to select the Weighting for the statistical calculations:

- **Time-weighted**

  Default. Weights each event value by the length of time over which it applies.

- **Time-weighted continuous**

Time-weights values, but does all interpolation between values as if the values represent continuous data, (standard interpolation) regardless of whether the attribute is stepped.

- **Time-weighted discrete**

  Time-weights values, but does all interpolation between values as if the values represent discrete, unrelated values (stair step plot) regardless of the attribute is stepped.

- **Event-weighted**

  Weights each event equally. This method requires at least one event in a time range (two events for standard deviation calculations). By default, events at the boundary of the calculation are handled as follows:

  - use events at both boundaries when there is only one calculation interval
  - include events at start time in multiple intervals and the intervals are in ascending time order
  - include events at the end time in multiple intervals and the intervals are in descending time order

- **Event-weighted - Exclude Earliest Event**

  EventWeighted, except that the event at the start time (earliest time) of an interval is not used in that interval.

- **Event-weighted - Exclude Most Recent Event:**

  EventWeighted, except that the event at the end time (most recent time) of an interval is not used in that interval.

- **Event-weighted - Include Both Ends:**

  Events at both ends of the interval boundaries are included in the event-weighted calculation.

Event horizon mode

Event horizon mode controls when events for attributes that support future data are returned in the PI AF data pipe. Possible values are described in the following table.

Value	Description
End of Stream	Events are returned as they created, without regard to the current time.
Time Offset	Events are only returned after they cross the relative time offset into the future from now. For example, with a time offset of zero, future tag data changes are not returned until their timestamp is less than or equal to the current time.

Value	Description
Time Offset with Outside Event	Events are only returned if they occur before the relative time offset into the future, or they are the next event that would cross this time horizon. For example, with a time offset of one hour, future tag data changes are not returned until their time stamp is less than or equal to the current time plus one hour, or they are the next known event after the one hour time horizon. This mode is useful for trending data into the future.

Restrictions on viewing time series data

PI point data reference configuration types

Although many configurations are possible with PI point data references, support differs for each data function. PI point data references can be grouped into the following types for time method configurations:

- **Simply configured PI point**

  All data functions are supported and there is no additional configuration. Time method is Automatic, Interpolated, or AtOrBefore.

- **Archive retrieval of specified PI point**

  Only functions in the interpolated values group (listed in the following table) are supported. Time method is Before, Exact, AtOrAfter, or After.

- **Relative time PI point**

  Simply configured PI point with relative time configured. Time method is Automatic, Interpolated, or AtOrBefore.

- **Time range only**

  Time method is not supported.

- **Time range with relative time**

  Time method is TimeRange or TimeRangeOverride.

The following table details which data functions are supported by the PI point data reference configurations listed above. These restrictions also apply to other client tools, such as PI DataLink, PI Vision (formerly known as PI Coresight), PI ProcessBook and PI WebParts.

Data function	Simply configured	Archive retrieval	Relative time	Time range only	Time range with relative time
`InterpolatedValue`	✓	✓	✓	✕	✓
`InterpolatedValues`	✓	✓	✓	✕	✓
`InterpolatedValuesAtTimes`	✓	✓	✓	✕	✓

Data function	Simply configured	Archive retrieval	Relative time	Time range only	Time range with relative time
`RecordedValue`	✓	✕	✓	✕	✕
`RecordedValues`	✓	✕	✓	✕	✕
`RecordedValuesAtTimes`	✓	✕	✓	✕	✕
`RecordedValuesByCount`	✓	✕	✓	✕	✕
`Async`	✓	✕	✓	✕	✕
`PlotValues`	✓	✕	✓	✕	✕
`Summary`	✓	✕	✓	✕	✕
`Summaries`	✓	✕	✓	✕	✕
`FilteredSummaries`	✓	✕	✕	✕	✕
`UpdateValue`	✓	✕	✓	✕	✕
`UpdateValues`	✓	✕	✓	✕	✕
`Annotations`	✓	✓	✓	✕	✕
`Filters`	✓	✕	✕	✕	✕
`AFDataPipe`	✓	✕	✕	✕	✕

Non-PI point data reference configuration types

Non-PI point data references can be grouped into the following types for time method configurations:

- **No data reference**

  Data reference is set to None.

- **Calculation-based data reference with time-series inputs**

  Calculation-based data references with time-series inputs, such as Analysis, Formula, String Builder, Table Lookup, and URI Builder support the base data functions. However, filtering and annotations functions are not supported. Calculation-based data references use recorded values of their inputs to generate timestamps for their own recorded values. If calculation-based data references contain no time-series inputs, they behave like the No-data-reference configuration type, except that they cannot be written to.

  > **Note:**
  > Table Lookup data references that are configured with a time column, as described in Table provided time series data, support the same data functions as the configuration type: Custom data reference generating its own time-series data.

- **Custom calculation-based data reference**

  Custom calculation-based data references that implement their own time-series calls are required to "opt-in" to gain access to base data functions.

- **Custom data reference generating its own time-series data**

  Custom data references that generate their own time-series data require code to be implemented for each data function call.

The following table details which data functions are supported by the non-PI point data reference configurations listed above. These restrictions also apply to other client tools, such as PI DataLink, PI Vision, PI ProcessBook and PI WebParts.

Data function	No data reference	Calculation-based data reference with time-series inputs	Custom calculation-based data reference	Custom data reference with its own time-series data
`InterpolatedValue`	✓	✓	OptIn[1]	Code[2]
`InterpolatedValues`	✓	✓	OptIn[1]	Code[2]
`InterpolatedValuesAtTimes`	✓	✓	OptIn[1]	Code[2]
`RecordedValue`	✓	✓	OptIn[1]	Code[2]
`RecordedValues`	✓	✓	OptIn[1]	Code[2]
`RecordedValuesAtTimes`	✓	✓	OptIn[1]	Code[2]
`RecordedValuesByCount`	✓	✓	OptIn[1]	Code[2]
`Async`	✓	✓	OptIn[1]	Code[2]
`PlotValues`	✓	✓	OptIn[1]	Code[2]
`Summary`	✓	✓	OptIn[1]	Code[2]
`Summaries`	✓	✓	OptIn[1]	Code[2]
`FilteredSummaries`	✗	✗	✗	Code[2]
`UpdateValue`	✓	✗	✗	Code[2]
`UpdateValues`	✓	✗	✗	Code[2]
`Annotations`	✗	✗	✗	Code[2]
`Filters`	✗	✗	✗	Code[2]
`AFDataPipe`	✓	✓	OptIn[1]	Code[2]

[1] Indicates that a custom data reference can be written with one method to declare which attributes it uses as inputs, and another method that transforms inputs to outputs. With input support, the AF SDK uses these methods to emulate the others, using a time series generated from input data.

[2] Indicates that code is needed for each method, if the time series comes from the system of record that is being connected to.

Organization of asset models in PI AF

The goal of your asset model organization is to make assets easy to find for your users. The main method of organization is the element tree. PI AF elements are organized in a tree structure. Individual elements can be organized and regrouped within the tree, without limitations. If you are just getting started with PI AF, then start by creating the element hierarchy.

You can additionally organize through categories to speed up and simplify browsing and searching information. Think of categories as labels that you can apply to PI AF objects. Each object can have multiple categories.

Note:
You can also organize assets within a process. See Process models in PI AF.

Topics in this section

- The structure of PI AF asset models
- The design of PI AF asset hierarchies
- Element references in the asset hierarchy
- Categorization of objects

The structure of PI AF asset models

PI AF objects are organized in a tree structure, similar to the file structure on a Windows computer. In Windows, rather than having thousands of files on your desktop, you typically group files under folders. The same concept applies to PI AF elements. Organizing elements into hierarchies makes navigation of the elements easier, and it also provides insights into how the elements relate to one another.

When you create an asset model, you need to decide on a structure that makes it easy for users to find the different assets. Consider who your users are and what they will be looking for. For example, maintenance engineers might want to use PI System Explorer to find and record maintenance information. For this audience you might want to group assets by equipment type.

For example, if you had three pump elements, you might create an element called Pumps and then place all the pump elements beneath it. If you had two elements representing tanks, you might put them all under a Tanks element.

Asset model organized by equipment type

On the other hand, if you have multiple plants in different locations, that same maintenance engineer might want to see all the equipment located at his own plant. The following illustration shows the same elements organized by plant.

Asset model organized by location

You are not restricted to only one organizational strategy. You can use an *element reference* to include the same asset in more than one place in the tree. For example, you could choose to organize by equipment type and by plant as well. In the following illustration, the hierarchy includes the geographical tree and the equipment tree side by side.

Mixed asset model

You could alternatively nest the equipment organization under the geographical organization.

The design of PI AF asset hierarchies

The goal is to create an asset (element) hierarchy that is going to make sense to the people who need to use it. If you have different types of users, then you might need more than one tree structure.

If you are just getting started, do not try to do everything at once. Create a hierarchy for a subset of your assets. For example, you might start by modeling all your tanks, or alternatively, your equipment in a single plant, or your equipment from a particular manufacturer. Another approach would be to create a hierarchy for a particular type of user.

Gather information:

- What assets will be included in the tree? In other words, what types of equipment do you want to model?
- Who is going to need to find assets in this tree? Maintenance engineers? Process control engineers? Operators? For each type of user, what tasks will they need to perform?
- What assets are important to each user and what types of information will they need? Consider asking a few representative users of each type about what data they need to access and how. This should inform your organizational strategy.

Again, start small. You might start with one type of user. For example, suppose maintenance engineers need to use the model. If you have several plants, each with a group of maintenance engineers responsible for the equipment at that plant, then you should probably include a tree that organizes equipment by plant.

From there, you might ask some maintenance engineers how they would want to access the equipment information. Perhaps they usually look for assets by equipment type but sometimes they need to search by manufacturer. You could create parallel trees, one organized by equipment type and another by manufacturer. Each asset would then appear in each tree. Or you could organize the tree by equipment type and then use categories to label each asset by manufacturer.

After you create the hierarchy for a type of user, you might have a few of them try it out for a period of time, then take their feedback to improve the hierarchy. You can always change a hierarchy.

Note:
When thinking about users, remember that the element hierarchy might also be exposed in certain PI client applications. Consider the users of those applications as well. For example, PI Vision exposes the tree for users searching for related assets. In PI Vision, related assets are elements built from the same template.

Videos

For information on how to create elements, watch this video:

https://www.youtube.com/watch?v=d-kPQGabJuY&rel=0

For information on how to build element hierarchies, watch this video:

https://www.youtube.com/watch?v=hpiQSzubDu8&rel=0

Element references in the asset hierarchy

You can use element references to add the same element to your element hierarchy more than once, as described in The structure of PI AF asset models. When you create a reference to an existing element, you essentially create a placeholder in the hierarchy that points back to the referenced element. This allows users to find the element in all the relevant contexts, but it does not create a copy of the element. The exact behavior depends on the reference type.

Topics in this section

- Reference types
- Create single element references
- Create multiple element references

- Locate other references to the same element
- Change reference types
- Creation of custom reference types

Reference types

When you add an element reference to an element hierarchy the exact relationship between the referenced element and the element to which you add it depends on the type of reference. PI AF provides three system-defined reference types:

- Parent-Child
- Composition
- Weak Reference

You can also create your own custom reference types, as described in Creation of custom reference types and Child templates.

Videos

For information on how to use reference types, watch this video:

https://www.youtube.com/watch?v=OL43oUyuFEI&rel=0

For information on how to build custom reference types, watch this video:

https://www.youtube.com/watch?v=VcXsWfUEYQ8&rel=0

For the conclusion of the reference type example, watch this video:

https://www.youtube.com/watch?v=Vwta8GG4Jgk&rel=0

Topics in this section

- Parent-Child reference type
- Composition reference type
- Weak reference type

Parent-Child reference type

The default reference type is parent-child. The reference type is strong since it allows the child element to have many parents and exist as long as it has one or more strongly related parents. When the last parent element of the child is deleted, the child itself is deleted. Use the parent-child reference type when you want the child element to exist as long as it has a strong reference to at least one parent, but you do not want the child element to be treated as a single unit with the parent element.

Effect of a deleted parent-child reference when that parent-child reference type exists elsewhere

Example

A meter belongs to a company and is attached to a building. You would use a parent-child reference between the `Company` element and the child `Meter` element, and another parent-child reference between the `Building` element and the child `Meter` element. If the reference between the `Meter` and the `Building` element is deleted, the `Meter` element continues to exist because it has a parent-child reference to the `Company` element. However, if the child `Meter` element is also deleted from the `Company` element, it no longer exists because all parent-child references have been removed.

Composition reference type

A reference type with a reference strength of composition means that the child element is really a part of the parent and does not exist without the parent. If you delete a parent element that has a child element that is compositionally referenced, you delete the child element also.

Effect of a deleted parent element on elements with a composition reference type

Example

Use a composition reference when the two objects in the relationship are considered one item. For example, a meter might be composed of two sensors, and so you would use a composition reference between the `Meter` element and each of its two child `Sensor` elements. When you delete the `Meter` element, the child `Sensor` elements are also automatically deleted.

Weak reference type

Use a weak reference between two elements when you want to create a relationship between two elements but you do not want that relationship to control the lifetime of the child element. For example, you may want to organize your meters into groups, but if all strong references to the meter are removed, then you want it to automatically be removed from the grouping. In this case, you would use a weak reference between the group parent element and the child meter element.

Effect of a deleted strong element on an element with a weak reference type

Example

A child element with a weak reference is deleted as soon as no more strong references exist. You may find this useful if you want one view of your assets to be the master view. For example, in the above illustration the master view might be the `Site ABC` view and contain elements with strong parent-child reference types. Other views, such as `Pumps`, reference elements with a weak reference type. If the strong `Pump01` element in the master `Site ABC` view is deleted, the weak-referenced `Pump01` element in the `Pumps` view is deleted automatically.

Create single element references

You create a single element reference to establish a hierarchical relationship between two elements.

Procedure

1. In the Elements browser, click the element that you want to reference and drag it to the appropriate parent element.
2. In the Choose Reference Type window, select the reference type.

   If you are not sure what reference type you want, you probably want the default reference type, parent-child. See Reference types for more information.
3. Click **OK**.

   The referenced elements are displayed in the browser. The referenced element icon indicates that they are references.

Video

For additional information on how to create an element reference type, watch this video:

https://www.youtube.com/watch?v=Q1kK71JrhLo&rel=0

Create multiple element references

You create multiple element references to establish a relationship between an element and other elements in the hierarchy.

Procedure

1. In the Elements browser, right-click an element and click **New** > **Add Element Reference**.

   Note that you can also select an element and drag it to the parent.
2. In the Add Element(s) window, click the **Find Multiple Elements** button.

3. In the Element Search window, select search criteria to locate the desired elements and click the **Search** button.

   The search results field shows the results of your search.

4. Select the elements that you want to reference and click **OK**.
5. In the Add Elements window, select the reference type. If you are not sure what reference type you want, you probably want the default reference type, parent-child. See Reference types for more information.
6. Click **OK**.

   The referenced elements are displayed in the browser. The referenced element icon indicates that they are references.

Locate other references to the same element

A single element can be referenced in multiple places in a hierarchy. You can find all the elements in the hierarchy that contain the element or a reference to the element.

Procedure

1. Select the element in the Elements browser.
2. In the **General** tab, click the **Parents** link.

   The element's parents are displayed in the Parents of *Element* window.

3. Click **Close**.

Change reference types

You can change the reference type of a child element and event frame that has at least one parent in the hierarchy and is not at the root level of the hierarchy (where the reference type must be parent-child).

Procedure

1. Choose from the following actions:

To ...	Do this ...
Change a child element reference type	a. In the navigator, select **Elements**. b. In the browser, expand the Elements tree and locate the child element with more than one parent. c. Right-click the child element and click **Convert** > **Change Reference Type**.
Change a child event frame reference type	a. In the navigator, select **Event Frames**. b. In the browser, expand the Event Frames tree and locate the child event frame with more than one parent. c. Right-click the child event frame and click **Convert** > **Change Reference Type**.

2. In the Change Reference Type window, select a parent from the **Parent** list.

3. Select a new reference type from the **New Reference Type** list. Only valid reference types for the selected parent are available.
4. To establish the selected parent as the primary parent (when the selected reference type is valid), select the **Primary Parent** check box.

   Note:
   To be valid, the selected reference type must be strong and no composition references can be defined for the element or event frame.

5. Click **OK**.

Creation of custom reference types

Custom reference types are easy to create. Each custom reference type must be based on one of the three pre-defined reference types. Through the use of custom reference types, you can create rules that allow PI AF to guide users when they create new elements. See Child templates for information on creating template references.

Create custom reference types

Custom reference types are created in child templates. You can also create them to define your own hierarchy of elements using a more specific relation. Custom reference types must be based on one of the pre-defined reference types.

Note:
You can create reference types more quickly in the Element Templates collection using referenced templates, as described in Create child template references.

Procedure

1. In the Library browser, right-click the **Reference Types** collection and select **New Reference Type**.
2. In the **Name** field, enter a name for the reference type.
3. In the **Child Name** and **Parent Name** fields, enter names for the child and parent.
4. Optional. In the **Categories** field, click and select one or more categories to which the reference type belongs from the Categorize window. You can also enter one or more reference type categories directly, separated by a semicolon.
5. In the **Reference Strength** field, select **Strong**, **Weak**, or **Composition** from the list. See Reference types for definitions of reference type strengths.
6. In the **Allowed Parent Element Template** and **Allowed Child Element Template** fields, select a template to which the reference type parent and child belong from the list.

Categorization of objects

PI System Explorer allows you to organize objects into categories. Categories are essentially object groups that you define yourself. Their purpose is to help you find objects more easily. When you search for an object, you can use the category as a filter to reduce the list of results. Define as many categories as you like. Objects can belong to multiple categories.

For example, suppose you have a set of elements representing tanks. Half of the tanks are manufactured by ACME company, and the other half are manufactured by EMCA company. To locate tanks by manufacturer, create an ACME category and an EMCA category.

Each object type has its own categories. You cannot apply categories from one object type to an object of another type. For example, you cannot apply an element category to a table. PI AF supports the following category types:

- Analysis
- Attribute
- Element
- Notification Rule
- Reference Type
- Table

Videos

For information on how to create categories, watch this video:

https://www.youtube.com/watch?v=ux3qiQVSePU&rel=0

For information on how to assign categories, watch this video:

https://www.youtube.com/watch?v=5wkvG-1jOgM&rel=0

For information on how to group by category, watch this video:

https://www.youtube.com/watch?v=A0OlJZ19qWA&rel=0

Topics in this section

- Create new categories
- Assign objects to categories

Create new categories

Create categories for specific object types. Alternatively, use PI Builder to create multiple categories in an Excel worksheet.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, under **Categories**, right-click on the category object type and select **New Category**.
3. In the *Object_Type* Category Properties window, enter a name for the category in the **Name** field.
4. Optional. Enter a description of the category in the **Description** field.
5. Click **OK**.
6. Optional. Click the **Security** link if you wish to set up custom access permissions to the category beyond those already established at the server, database, or collection level. For more information, see Configure security for objects.

Assign objects to categories

Assign objects to the categories that you have created for each category type.

Procedure

1. In the viewer, use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL+<click>) objects that you want to categorize.
2. Right-click and select **Categorize**.
3. In the Categorize window, select the category or categories to assign to the selected objects.
   a. If the category you need is not listed, click the **New Category** button.
   b. In the Attribute Categories Properties window, create the new category and click **OK**.
4. Click **OK**. The categories are applied and are displayed in the **Categories** field in the configuration panel for each object.
   a. If any selected objects are created from templates, you can modify the template or not add the category to that particular object.

   Choose one of the following options:

   - Click **Modify the template and add the category** to modify all attributes derived from the template with the category.
   - Click **Do not add the category** to prevent the category from being assigned to the object.

   b. To apply the option to other selected attributes that are derived from templates, select the **Apply choice to remaining templates** check box.
   c. Click **OK**.

Representation of assets in PI AF

Each physical object that you include in your model is represented by a PI AF element. Physical objects are typically pieces of equipment, such as pumps or tanks. To associate data with an element, you create attributes on the element. Attributes can hold simple values representing fixed information, such as the diameter of a tank. An attribute can alternatively reference a PI point, a formula, a value from an external relational database, and more.

Templates

You can create each element individually, but OSIsoft recommends that you base individual elements on an asset template that represents the type of equipment. Element templates enable you to:

- Configure the template once; no need to individually configure each element based on the template.
- Update element structure across all elements belonging to the template.
- Maintain consistency in the asset model.
- Enable powerful features in PI AF client applications.

Topics in this section

- Asset representation with templates
- Template strategy
- Element templates
- Child templates
- Asset representation with elements
- Storage of application-specific information

Asset representation with templates

PI AF allows you to base similar objects on a single template. Templates essentially define a set of base attributes for all the objects that use that template. Create the template once and you can create as many elements based on the template as needed. If you make a change to a template, the change is automatically reflected in all the elements that use that template.

For example, suppose you have 100 pumps with the same three attributes. You can create a template for one pump and then base all the other pumps on that one template. The attributes in the template are automatically created for you in the pumps that are based on that template.

Now, suppose you need to make a change to the pump objects. You simply make the change in the template, and PI AF automatically propagates the change to all the pump objects that are

based on that template. Templates are a powerful tool, not only for creating new objects, but for keeping existing objects consistent and up to date.

A further advantage is that visualization tools can provide special features for objects based on templates. For example, suppose you build a trend for a pump based on a template. A visualization tool might let you swap or add in any other pump that is based on the same template. Assets based on the same template are sometimes called *related assets*.

Note:
You can also create templates for event frames, transfers, models, and notifications.

Template strategy

In almost every case, it is best to base your elements on templates. You not only save time but also ensure that you have consistent definition across all of the elements based on that template. Any changes that you make to an element template are propagated to every element that is based on that template. A single template modification can alter hundreds of elements. This allows you to make changes to your model in a single place; you don't need to update every element.

You do not have to plan and create all your templates at once. A good approach is to start by modeling a single type of asset. Create a template for the asset type. Decide what data, calculations, and other properties you need to store for that type of asset. Each of these items requires an attribute template.

Base templates

Template usage can be very broad or very specific. A template can define a specific type of measurement device, such as a brand-name instrument, or it can be a broad-use template specifying a particular role, such as a liquid mass meter. Depending on how broadly you define the template, you might find that the list of attributes is slightly different for different subsets of assets. In this situation, consider using a base template (for more information, see Base and derived templates). The alternative is to use a different template for each asset subset.

Extensions

You can also set up element templates to include attributes (as well as ports) that are not defined or maintained by the template (for more information, see Extensions). You can add simple, asset-specific attributes without the need to add them on all instances of the template.

Note:
Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.

Element templates

An element template is essentially a model of a type of asset. Element templates make creating displays, notifications, calculations, and analytics much simpler because equipment of the same type can share the implementation. Typically you create an element template that represents a type of equipment, such as a tank or pump. You can easily create elements based on the template because most of the element configuration is defined in the template.

When you change an element template, those changes propagate to all elements derived from it.

Deleting an element template deletes any notification or analysis templates that target it.

Note that:

- An element derived from a template gets its definition of categories, attributes, and ports from the template.
- If the template allows extensions, then a derived element can contain extra categories, ports, and attributes that are not defined in the template.
- Element templates can specify the allowed parent and child references (Parent Reference Types and Child Reference Types) for an element created from the template. This restricts the allowed relationships between elements in the hierarchy.

Video

For information on how to create element templates, watch this video:

https://www.youtube.com/watch?v=VYWXctstVg0&rel=0

Topics in this section

- Default attribute
- Base and derived templates
- Extensions
- Naming patterns
- Create element templates
- Find template references
- Define templates for other objects

Default attribute

The default attribute is an attribute that is most representative of the element. For example, if a user were to reference a tank, perhaps `Level` would be the single best attribute to use. For an electrical meter, the best attribute to use could be `Voltage`. The attribute must be at the top level of the attribute hierarchy of the element.

Notifications and asset analytics can take advantage of the default attribute to simplify some tasks. For example, if you were to create a comparison between two electrical meters, and had not specified which attributes to compare, the comparison would be performed using the `Voltage` attribute of the meters because you had made `Voltage` the default attribute.

Base and derived templates

A base template is best used when you are modeling elements that share most attributes, but have a few attributes that differ. A base template passes on all its attributes (as well as ports and analyses) to any derived templates that are created from it. When you later create an element from such templates, that element contains not only every attribute (as well as ports and analyses) from the base template, but also those from the derived template.

Base template example

Suppose you have a set of tanks, some with two valves and some with one valve: you can create an element template for the one-valve models, and use that as the base template for the derived template that you create for the two-valve models. When you create objects based on the two-valve model, they inherit all the attributes from the one-valve model, as well as any additional attributes defined for the two-valve model.

Derived templates

You define derived templates from a base template. You can set the base template by selecting from a **Base Template** list in the **General** tab of the element template you are defining, or you can right-click on an element template that is already displayed in the Library browser and click **New** > **New Derived Template**. The new template is immediately added to the Library browser with a new name ***Base_Template*Derived1**. You can then refine the definition, including the name, as described in Create element templates.

Override of base-template attribute values

Situations may exist where you need to use attribute template values in a derived template that suppress or override attribute template values inherited from their base template. Consider the following situations:

- Different substitution patterns are needed for PI points on the same type of asset (such as trucks with a common set of measurements, but truck manufacturer A uses a different PI point naming pattern than manufacturer B).
- Some assets employ direct measurement for an attribute PI point, whereas others may need that attribute value to be calculated. Similarly, an attribute value may be calculated for both types of assets, but the calculation is slightly different between the two.
- Different default UOMs or categories are needed for the same attribute, based on template.
- Not all attribute templates in a derived template are applicable (for example, some meters may only need three of the five attributes in a generic base `Meter` template).

You override base template attribute values by creating an attribute template with exactly the same name in the derived template. You can then modify properties as needed.

You can also prevent an attribute template that has been inherited from a base template from being visible to users or a client application, such as PI Vision, by creating an attribute template of the same name in the derived template and selecting **Excluded** as its **Properties** setting. For more information, see Excluded attribute property.

Template inheritance

By default, element templates are arranged by name in the Library browser. To determine the relationship that exists between templates, you might prefer to arrange them by template inheritance instead. You can right-click the **Element Templates** collection and click **Arrange By** > **Arrange By Template Inheritance**. Expand templates with ⊞ beside them to reveal derived templates.

Extensions

You can allow some attributes to be defined in the element itself, independent of the template. To do this you configure the template to allow extensions. Attributes that are created as extensions do not change when the template changes.

> **Note:**
> If you have a template that allows extensions and you later change it to disallow extensions, no new extended element attributes can be added to elements based on that template. However, all existing extended element attributes remain.

Extensions permit flexibility in cases where assets are similar in many areas, but a number of small variations exist. For example, suppose you build a template for a specific model of a car. All the cars of this model have the same core set of features: four tires, a steering wheel, and so on. However, one car may have a spoiler while the other car may not. One might have air conditioning, while another does not. Extensions are intended to handle this type of variation.

When extensions are enabled, you can base all the elements on a template, while adding additional information to each element as required. However, if you have multiple elements that are very similar to the template definition, but all require the same extra attributes, a base template might be a better approach than allowing extensions.

> **Note:**
> Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.

Naming patterns

When you define naming patterns for element, event frame, transfer, and model templates, you can use the substitution parameters described in the following table. These naming pattern shortcuts are available in addition to the substitution parameters that are listed in the **Naming Pattern** field when you click ▸.

Naming pattern substitution parameters

Parameter name	Substitution
Description parameters	
%DESCRIPTION:*path***%**	The object's description represented by the path to the object. If the path is not specified, the description of the current object is used.
%\ELEMENTDESCRIPTION%	The description of the object's root element, or the description of the primary referenced element of the object's root event frame.
%..\ELEMENTDESCRIPTION%	The description of the object's parent element, or the description of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENTDESCRIPTION%**, to retrieve further ancestors.
%\EVENTFRAMEDESCRIPTION%	The description of the object's root event frame.
%..\EVENTFRAMEDESCRIPTION%	The description of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAMEDESCRIPTION%**, to retrieve further ancestors.
ID parameters	
%ANALYSISID%	The ID of the object's analysis.
%ATTRIBUTEID%	The ID of the object's attribute or attribute template.
%|ATTRIBUTEID%	The ID of the object's root attribute or attribute template.
%..|ATTRIBUTEID%	The ID of the object's parent attribute or attribute template. Use **..|** notation, such as **%..|..|ATTRIBUTE%**, to retrieve further ancestors.

Parameter name	Substitution
%DATABASEID%	The ID of the object's PI AF database.
%ELEMENTID%	The ID of the object's element, or the ID of the primary referenced element of the object's event frame.
%\ELEMENTID%	The ID of the object's root element, or the ID of the primary referenced element of the object's root event frame. Use **..** notation, such as **%..\..\ELEMENTID%**, to retrieve further ancestors.
%..\ELEMENTID%	The ID of the object's parent element, or the ID of the primary referenced element of the object's parent event frame.
%EVENTFRAMEID%	The ID of the object's event frame.
%\EVENTFRAMEID%	The ID of the object's root event frame.
%..\EVENTFRAMEID%	The ID of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAMEID%**, to retrieve further ancestors.
%ID:*path***%**	The object's ID represented by the path to the object. If the path is not specified, the ID of the current object is used.
%MODELID%	The ID of the object's PI AF model.
%SYSTEMID%	The ID of the object's PI AF server.
%TRANSFERID%	The ID of the object's PI AF transfer.
Name parameters	
%ANALYSIS%	The name of the object's analysis.
%DATABASE%	The name of the object's PI AF database.
%\ELEMENT%	The name of the object's root element, or the name of the primary referenced element of the object's root event frame.
%..\ELEMENT%	The name of the object's parent element, or the name of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENT%**, to retrieve further ancestors.
%\EVENTFRAME%	The name of the object's root event frame.
%..\EVENTFRAME%	The name of the object's parent event frame. Use **..** notation, such as **%..\..\EVENTFRAME%**, to retrieve further ancestors.
%NAME:*path***%**	The object's name represented by the path to the object. If the path is not specified, null is returned since it references the name of the current object.
%SYSTEM%	The name of the object's PI AF server.
%\TEMPLATE%	The name of the root template of the object's element or event frame.
%..\TEMPLATE%	The name of the parent template of the object's element or event frame. Use **..** notation, such as **%..\..\TEMPLATE%**, to retrieve further ancestors.
Path parameters	
%ELEMENTPATH%	The path of the base element, the element of an attribute, or the primary referenced element of an event frame.
%..\ELEMENTPATH%	The path of the object's parent element, or the path of the primary referenced element of the object's parent event frame. Use **..** notation, such as **%..\..\ELEMENTPATH%**, to retrieve further ancestors.

Parameter name	Substitution
%EVENTFRAMEPATH%	The path of the event frame or the element of an attribute of an event frame.
%..\EVENTFRAMEPATH%	The path of the object's parent event frame. Use `..\` notation, such as **%..\..\EVENTFRAMEPATH%**, to retrieve further ancestors.
Time parameters **Note:** You can specify a format string with all time substitutions. The format string is separated from the substitution name by a colon. For example, `%TIME:d%` uses the short date pattern, whereas `%UTCTIME:yyyy/MM/dd HH:mm:ss.fff%` specifies the format in full.	
%ENDTIME:*yyyy-MM-dd HH:mm:ss.fff***%**	The current local end time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters.
%UTCENDTIME:*yyyy-MM-dd HH:mm:ss.fff***%**	The current Coordinated Universal Time (UTC) end time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters.

Create element templates

Create and configure an element template to ensure consistent definitions for elements in your asset structure.

Procedure

1. In the Library browser, right-click the **Element Templates** collection and click **New Template**.
2. Adjust settings on tabs to configure the element template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing template. Select the base template from the list. For more information on base templates, see Base and derived templates.
Type	You can base the template on an existing template. Select the base template from the list. For more information on base templates, see Base and derived templates.
Categories	Optional. You can organize objects by grouping them into categories. To browse available categories or to create a new category, click . See Categorization of objects.
Default Attribute	Optional. Select a default attribute from the drop-down list. See Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%TIME:yyyy-MM-dd HH:mm:ss.fff%** The current local time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute, represented by the path. You can also use other naming pattern substitution parameters. For example, if you want an element path included in the naming pattern, enter `%ELEMENTPATH%` as a substitution parameter. For a complete list of naming pattern substitution parameters, see Naming patterns. Each element derived from the template will have a unique, identifiable name. To ensure that new elements created from the template have an incremental number, enter * at the end of any pattern you enter here. If left blank, PI System Explorer uses the element template **Name** field and adds an asterisk to add an incremental number as needed. Some substitution parameters may not be applied when a derived element is created. To ensure a derived element's name fully reflects the naming pattern, right-click the element and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters.
Allow Extensions	Select this checkbox to enable additional attributes (as well as ports) to be defined for an element, beyond those defined in its template. See Extensions. **Note:** Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Annotations	Click this link to add one or more annotations to an element. For more information, see Element annotations.
Location	Click this link if you wish to set up location attribute traits for the element. However, you can also set them on an attribute if you prefer. Note that you can only assign one set of location attributes per element. For more information, see Attribute traits.

Option	Description
Security	Click this link if you wish to set up custom access permissions to the element template beyond those already established at the server and database level. For more information, see Configure security for objects.

3. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.
4. If you are creating a model, use the **Ports** tab to specify ports, which define end-points for connections between elements within a model. See Process models in PI AF.
5. Optional. Use the **Analysis Templates** tab to create an analysis template for the element template. For more information, choose from the following:
   - To create an expression analysis template, see Create an expression analysis template.
   - To create a rollup analysis template, see Create a rollup analysis template.
   - To create an event frame generation analysis template, see Create an event frame generation analysis template.
   - To create an SQC analysis template, see Create an SQC analysis template.
6. Optional. Use the **Notification Rule Templates** tab to select or create a notification rule template for the element template. For more information, see Create a notification rule template.
7. To save your work, press CTRL+S or click **Check In**.

Find template references

Locate where a template is used in PI System Explorer.

Procedure

1. In the Library browser, choose from the following options:
   - To locate an element template, expand **Templates** > **Element Templates** and select a template.
   - To locate an event frame template, expand **Templates** > **Event Frame Templates** and select a template.
2. Choose from the following options:

To find ...	Do this ...
Derived templates	Click **Derived Templates**. The Find Derived Templates for *Element_Template* window displays all element templates that have the selected element template as their base template. The list also includes element templates that are indirectly derived from the selected template, through multiple templates.

Elements created from the template	Click **Elements**. The Find Elements for *Element_Template* window displays all elements, models, transfers, and event frames that have been created directly from the selected element template. **Note:** To display the full path to elements, right-click on the column header (or below the search results grid) and click **Show Full Paths**.
Elements derived from the template	Click **Derived Elements**. The Find Derived Elements for *Element_Template* window displays all elements, models, transfers, and event frames that have been created from the selected element template, as well as from any template that is derived from it. **Note:** To display the full path to derived elements, right-click on the column header (or below the search results grid) and click **Show Full Paths**.
Referenced parent templates	Click **Referenced Parent Templates**. The Find Referenced Parent Templates for *Element_Template* window displays element templates that have specifically been linked to this element template as a parent through a reference type. The list of templates includes those that indirectly have been linked through template inheritance.
Referenced child templates	Click **Referenced Child Templates**. The Find Referenced Child Templates for *Element_Template* window displays element templates that have specifically been linked to this element template as a child through a reference type. The list of templates includes those that indirectly have been linked through template inheritance.
Event frames created from the template	Click **Event Frames**. The Find Event Frames for *Event_Frame_Template* window displays all event frames that have been created directly from the selected event frame template. **Note:** To display the full path to event frames, right-click on the column header (or below the search results grid) and click **Show Full Paths**.

Event frames derived from the template	Click **Derived Event Frames**. The Find Derived Event Frames for *Event_Frame_Template* window displays all event frames that have been created from the selected template, as well as from any template that is derived from it. **Note:** To display the full path to derived event frames, right-click on the column header (or below the search results grid) and click **Show Full Paths**.

Define templates for other objects

In addition to element templates, define templates as needed for event frames, models, and transfers. These templates have all the necessary attributes and properties for the role of the object. You then create instances of the object by basing them on the configured template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, open **Templates** to display the following object collections:
   - **Element Templates**
   - **Event Frame Templates**
   - **Model Templates**
   - **Notification Templates**
   - **Transfer Templates**
3. Right-click an object collection and select **New Template**.

   The new template displays the following tabs:
   - **General**
   - **Attribute Templates**
   - **Ports** (not available for event frame templates)
   - **Analysis Templates** (element templates only)
4. Fill in these tabs just as you would for an element template. See Create element templates.
5. To save your work, press CTRL-S or click **Check In**.

Child templates

In some cases, you might want to create a template that has one or more child templates. For example, suppose you have a template representing a motor and you want a separate template for the shaft. You want the shaft template to be a child of the motor template. You cannot directly create a child template in the same way that you would create a child element. Instead, you create a new referenced template below the `Motor` template.

This also creates a new reference type in the Reference Types collection.

Now, suppose you create an element based on the `Motor` template. PI System Explorer does not automatically create the child element, `Shaft`. However, when you manually create a child element on the motor element, the `Motor-Shaft` reference type is added to the list of reference types and the `Shaft` template appears in the Element Template list.

Create child template references

Create a child template reference when you need to establish parent-child relationships between two element templates.

Procedure

1. In the Library browser, right-click the template to which you want to add the child template reference and click **New** > **New Referenced Template**.
2. In the **Set Referenced Element Template Name** window, type a name for the child template, and click **OK**. This creates a child template in the **Templates** collection and a new reference type in the **Reference Types** collection.
3. Optional. To define a reference type other than strong parent-child, select the **Edit reference type** check box.
   a. In the Reference Type Properties window, select the desired reference strength from the **Reference Strength** list.
   b. Click **Check In**.
   c. Click **OK** to close the window.
4. Complete the element template definition, as described in Create element templates.

Asset representation with elements

In your asset model, elements represent physical objects, such as pumps or tanks. Each element has associated attributes that store properties and data for that element. Typically, you base each element on a template. For example, you would base each pump element on a pump template.

Topics in this section

- Create elements based on a template
- Create elements not based on a template

- Find where an element is used
- Deletion of elements
- Attribute-value reset to original properties
- Convert elements to element templates
- Change element templates

Create elements based on a template

Use the element templates that you have defined to create elements in your asset hierarchy.

Procedure

1. In the Elements browser, right-click the collection of elements or an individual element and click **New Element**.
2. In the **Choose Element Template** window, select an element template in the **Element Template** field.
3. In the **Add child element using the reference type** field, select a reference type from the list of available types.

   The available reference types depend on the selected element template and what reference types are defined in the database. If you are not sure what reference type to use, select the default reference type, parent-child.
4. Click **OK**.

   Property tabs for the new element are displayed in the viewer. Because the element is based on a template, most fields on the **General** tab are read-only (with gray or hash-pattern shading).
5. In the **Name** and **Description** fields, type a name and description for the element.
6. Optional. Perform the following actions as needed.
   - Click the **Extended Properties** link to create properties for client applications such as PI ProcessBook and PI WebParts. For more information, see Storage of application-specific information.
   - Click the **Location** link to set up location attribute traits for the element. For more information, see Set location attribute traits.
   - Click the **Annotations** link to enter a note for the element. For more information, see Element annotations.
   - Click the **Security** link to set up custom access permissions for the element. For more information, see PI AF object security.
7. Optional. Click the **Attributes** tab to review the attributes defined by the template. You cannot add new attributes unless the template allows extensions.
8. Optional. Click the **Notification Rules** tab to review any notification rules defined by the template. You cannot add new notification rules unless the template allows extensions.
9. To save your work, press CTRL-S or click **Check In**.

Create elements not based on a template

Although you can create elements that are not based on a template, OSIsoft recommends that eventually you base your elements on templates so that you can take advantage of powerful template-based features available in client applications. You can use the Convert to Template feature, as described in Convert elements to element templates.

Procedure

1. In the Elements browser, right-click the collection of elements or an individual element and click **New Element**.
2. In the **Choose Element Template** window, select a reference type in the **Add child element using the reference type** field.

   The available reference types depend on the element template and what reference types are defined in the database. If you are not sure what reference type to use, you probably want the default reference type, parent-child.
3. In the **Element Template** field, click **<None>**.
4. Click **OK**.

   Property tabs for the new element are displayed in the viewer.
   - Use the **General** tab to perform the basic element configuration.
   - The **Child Elements** tab is mainly used to view existing child elements on an existing element. You can optionally create new child elements directly from this tab.
   - Use the **Attributes** tab to create an attribute for each property or data item.
   - Use the **Ports** tab only if you are modeling a process.
   - Use the **Analyses** tab to select an analysis for the element.
   - Use the **Notification Rules** tab to select a notification rule for the element.
   - Use the **Version** tab to add version information for the element.
5. Optional. Perform the following actions as needed.
   - Click the **Extended Properties** link to create properties for client applications such as PI ProcessBook and PI WebParts. For more information, see Storage of application-specific information.
   - Click the **Annotations** link to enter a note for the element. For more information, see Element annotations.
   - Click the **Security** link to set up custom access permissions for the element. For more information, see PI AF object security.
6. To save your work, press CTRL-S or click **Check In**.

Find where an element is used

Use the **Find** links on the **General** tab for a selected element to locate where and how it is used in the asset model.

Procedure

1. Select an element in the Elements browser.
2. Choose from the following options:

To find ...	Do this ...
Parent elements	Click the **Parents** link. The Parents of *Element* window opens.
Child elements	a. Click the **Children** link. b. In the Element Search window, enter element name text in the search field. To refine your search, you can enter additional criteria in the **Criteria** section. c. Click **Search**. Matching elements are displayed in the **Results** section.
Event frames connected to the element	Click the **Event Frames** link. ◦ If a match is found, the Find Event Frames for *Element* displays a list of all event frames that reference the element. ◦ If no match is found, a `No Event Frames Found` message is displayed.

Deletion of elements

When you delete an element in PI AF 2.4 and later, PI AF automatically deletes the notifications and analyses that target that element. If you wish to repurpose a notification that is attached to an element that you plan to remove, ensure you do so before you remove the element.

Alternatively, you can change the default behavior so that PI AF does not automatically delete notifications and analyses when you delete the targeted element.

You use the **afdiag** command line utility with the **/EnablePropagationOfTargetDeletion** parameter:

```
afdiag /EnablePropagationOfTargetDeletion-
```

You should be aware that disabling the setting can cause problems, such as:

- Notifications are left open (and remain active)
- New notifications are not created
- Expected emails are not sent

Attribute-value reset to original properties

When you have created elements based on templates in a test environment and later want to transfer your work to a production environment, you can reset all attributes for an element to the original properties as defined by a template. You can right-click an attribute for an element and click **Reset to Template**.

For example, suppose you configure a PI AF attribute to use the name of PI Data Archive as part of the attribute's configuration in a test environment. You might want this to reflect the new name for PI Data Archive when you transfer your work to a production environment.

Convert elements to element templates

OSIsoft recommends that elements be based on templates. If you created elements and attributes and did not originally base them on a template, you can later convert them to a template.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, locate the element that you want to convert to a template.
3. Right-click the element and click **Convert** > **Convert to Template**.

   A template named ***ElementName*Template** is created and displayed in the **Template** field on the **General** tab.
4. Optional. To rename the template, click **Library** in the navigator.
   a. In the Library browser, locate the new template in the **Element Templates** branch.
   b. Right-click the template and click **Rename**.
   c. Modify the default name as needed. The new name must be unique across all templates.

Change element templates

You can change the template on which an existing element is based, or add a template to an element that is not currently based on a template.

Before you start

Exercise caution when you change an element template, since there can be unintended consequences. For example, attributes could be deleted if they were defined by the old template and are not present in the new template. Be sure that attributes with the same name in both the old and new template have the same data type.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, right-click the element that you want to change and click **Convert > Change Template**.
3. In the Choose Element Template window, select a template from the **Element Template** list.
   a. Optional. To display only templates in a specific category, select a category from the **Templates of category** list.
4. Click **OK**.

Storage of application-specific information

Extended properties are properties that other applications define on PI AF objects. For example, PI WebParts stores Icon and URL in a PI AF element's extended properties. Applications typically make use of the information stored in the Extended Properties window programmatically with AF SDK.

An **Extended Properties** link is displayed on the **General** tab of many object types in PI System Explorer. This link can be used to configure one or more additional properties for an object.

> **Note:**
> Users do not need to use this advanced PI AF feature.

Create extended properties

In general, client applications such as PI WebParts and PI ProcessBook use the Extended Properties feature to write properties, and users do not need to use this advanced feature.

Procedure

1. Click the **Extended Properties** link.
2. In the Extended Properties window, choose from the following actions:
   - To create the first extended property, click the **New Extended Property** link.
   - To create an additional extended property, right-click in the **Name** column and select **New Extended Property**.
3. In the **Name** field, delete the default text and enter the extended property name.
4. Click in the **Value** field and choose from the following options:
   - Enter a string. To visualize the text you are entering, press F2 and type in the Text Visualizer window.
   - Click T and select from the list of **Basic Types**, **Array Types**, and **Objects**.
     - If you choose an object such as **Attribute** or **Element**, click ... and locate the desired object.
     - If you choose **File**, click ▸ and select **Upload** to locate the desired file.
5. Click **Close**.

Association of data with PI AF assets

To associate data with an asset, you create attributes on the element that represents that asset.

Attributes can hold simple values that represent fixed information, such as the diameter of a tank. They can also hold values from PI points, be derived from formulas, or hold data from outside the PI System through the use of PI AF tables.

Enumeration sets

If you need an attribute to hold only predefined values, you can use an enumeration set. An enumeration set maps an ordered set of user-defined constant values to a set of strings. You can use the strings to provide brief, descriptive text to use within your PI AF model.

Videos

For information on how to create element attributes, watch this video:

https://www.youtube.com/watch?v=pLmIPOOIcEw&rel=0

For information on how to create child attributes, watch this video:

https://www.youtube.com/watch?v=BiXA13ptVTM&rel=0

Topics in this section

- Attribute templates
- Enumeration sets

Attribute templates

Attribute templates are associated with element templates. Just as an element template represents a type of asset, an attribute template represents a type of data configuration. When you create an instance of the element template, that element contains an attribute for each attribute template. These attributes inherit all properties configured on the attribute template.

Rather than create attributes on each element, OSIsoft recommends that you create attribute templates in an element template. Whenever you create elements based on that template, PI AF automatically creates the attributes for you. You still need to give each attribute a value.

Modification of attribute descriptions created from attribute templates

Beginning with 2017 R2, you can modify the descriptions of attributes that are created from a template. You may find this helpful when differentiating between attributes that use the same attribute template. For example, in a series of pumps, each pump has a `Flowrate` attribute with a PI point data reference. The template description for `Flowrate` is simply `Pump flowrate`. To make that description unique for each instance of the pump, you could use PI Builder to load the **`descriptor`** attribute for each pump flowrate PI point and incorporate that value into each `Flowrate` description, thereby customizing it.

To modify descriptions created from an attribute template, you must use a PI AF client and server that are running PI AF 2017 R2 or later. Older clients that connect to a PI AF server running 2017 R2 or later continue to see only the description from the attribute template.

> **Note:**
> Because extra memory is used to store descriptions for attributes that have been overridden, loading them into client applications can take a little longer.

Video

For information on how to create attribute templates, watch this video:

https://www.youtube.com/watch?v=dCx5_Aw5x24&rel=0

Attribute properties

You need to set up the properties for each element attribute. After you assign properties in an attribute template, users can only modify the **Exclude** property on individual attributes.

Topics in this section

- Configuration Item attribute property
- Excluded attribute property
- Hidden attribute property
- Indexed attribute property
- Manual Data Entry attribute property

Configuration Item attribute property

You assign the **Configuration Item** property to an attribute that represents inherent properties of an asset. Any attribute that has a constant value can be a configuration item. Because the attribute value of a configuration item is presumed to be constant, PI System Explorer automatically checks out the attribute when you change the attribute value. To commit the change, you need to check it in. In PI System Explorer, configuration attributes are marked with a pencil icon (✎).

Attributes with values that change, such as PI point data references, or formula data references, should not be configuration attributes. When you change the value of a non-configuration attribute, PI System Explorer does not check out the attribute. For example, suppose you have an element representing a tank. The element has the following attributes:

- ✎ manufacturer
- ✎ photo (file)
- ✎ serial number
- ✎ maximum temperature rating
- ✎ maximum volume
- temperature (PI point data reference)
- volume (PI point data reference)

  The first five attributes would typically be configuration items, since these values will not change unless you change the equipment. The last two attributes would not be configuration items, since the values change all the time.

Excluded attribute property

You exclude an attribute in situations where not all attributes in an element template apply. For example, suppose you have five attribute templates defined in an element template named `Meter`, which you use to create elements representing several different types of meter in use at your facility. In some kinds of meter (`meterA`), only three attributes from the `Meter` template are applicable, whereas in the remainder (`meterB`) all five attributes apply. In meters of type A, you should select the **Excluded** property for the two attributes that are inapplicable. For all practical purposes, those two attributes "seem" nonexistent to most PI AF client applications since searches retrieve the digital state value `No Result`.

> **Note:**
> An excluded attribute cannot be used to return elements and event frames that are referenced by an excluded attribute value.

When the **Excluded** property is assigned in an attribute template, the property is indicated by beside the attribute name if the **Show Excluded Attributes** option is enabled. Attributes that are derived from that template appear on an element **Attributes** tab with in the Template column and **Excluded** in the Value column. You can toggle the display of such attributes with the **Hide Excluded Attributes** and **Show Excluded Attributes** command on the **Attributes** tab context menu. Note that when **Show Excluded Attributes** is enabled, users can change the **Excluded** property setting for an attribute from the setting it is assigned in an attribute template.

Alternatively, you can clear the **Show Excluded Attributes** option in **Tools** > **Options** to hide excluded attributes everywhere in PI System Explorer. When attributes are hidden, the `Excluded attributes are hidden` message is displayed on the **Attributes** tab.

Hidden attribute property

A hidden attribute is one that cannot be retrieved in searches from PI AF client applications such as PI Vision. This property is useful if an attribute is being used to hold an intermediate result, such as a table lookup result that can then be retrieved by a PI point data reference, or is being used solely to populate a PI point name in a substitution parameter.

It can also be useful to set an attribute property to **Hidden** in a template when configuration has not been fully completed in the element itself. For example, elements are being created from a template with a PI point data reference, but because some instrumentation is missing, PI points do not yet exist for all the elements. By setting the attributes for the missing instrumentation to **Hidden**, a PI AF client application is prevented from obtaining an error result from a search.

When the **Hidden** property is assigned to an attribute in an attribute template, the property is indicated by in the Hidden column beside the attribute name. The same icon is displayed next to the attribute name on the **Attributes** tab when the template is assigned to an element.

Indexed attribute property

You index an attribute to optimize it for fast search results and fast value retrieval. Indexes only improve performance for attributes whose values are stored in the PI AF database. This means that you do not gain any benefit from indexing attributes that obtain their values from data references, as is the case with PI points and linked tables. For event frames, however, when

attribute values are captured they are written to the PI AF server and as a result can benefit from indexing.

> **Note:**
> Only the first 40 characters of indexed attributes with a **String** value type are used in searches.

When you index an attribute, both the attribute and the attribute value are returned more quickly in a search. However, indexed attributes increase the table size in your PI AF database and they create some CPU load on the SQL server. Indexed attributes are marked with the **i** icon.

You typically index an attribute that you think users are likely to search for frequently. For example, if you know that you want to locate assets by searching for a serial number, you should index the serial number attribute.

Manual Data Entry attribute property

You flag an attribute for manual data entry to enable client applications (such as PI Vision and PI Manual Logger) to guide users to enter data for that attribute. For example, in a future release of PI Manual Logger, this property will identify tags in a tour that operators should collect manually, in contrast to tags that are collected automatically via an interface or connector.

To determine which attributes have this property, add a Manual Data Entry column to an attribute grid in the **Elements** viewer, where attributes with this property display as **True**.

Attribute traits

Attribute traits identify commonly held characteristics or related behaviors of a parent attribute. PI AF supports attribute traits for limit conditions, forecasted values, geolocation information, reason codes, asset health, and analysis start triggers. Users and client applications like PI Vision can then find, display, and analyze such related attributes more effectively.

To set and access attribute traits, both the PI AF client and server need to be running PI AF 2016 or later.

Limit attribute traits

You use limit attribute traits to identify the expected range of a process variable or a measurement attribute. Limit attribute traits are often set elsewhere (for example, on a control system), or represent inherent properties of the parent attribute, such as temperature. A typical usage would be to set up conditions for an alarm. PI Vision, for example, uses most limit attributes as input when configuring multi-state behavior for a visual alarm.

Limit attribute traits are child attributes of the measurement they describe and have the same value type and UOM as the parent. The only exception is when the value of a limit attribute trait is obtained from a data reference, in which case the source UOM of a defined data reference can be different. You can also only set limit attribute traits for parent attributes that are numeric. Note that you can only assign a specific limit attribute trait once to a parent attribute.

The following limit traits are defined:

Limit attribute trait	Description
Minimum	Indicates the very lowest possible measurement value or process output.
LoLo	Indicates a very low measurement value or process output, typically an abnormal one that initiates an alarm.
Lo	Indicates a low measurement value or process output, typically one that initiates a warning.
Target	Indicates the aimed-for measurement value or process output.
Hi	Indicates a high measurement value or process output, typically one that initiates a warning.
HiHi	Indicates a very high measurement value or process output, typically an abnormal one that initiates an alarm.
Maximum	Indicates the very highest possible measurement value or process output.

Video

For information on how to set limit attribute traits, watch this video:

https://www.youtube.com/watch?v=jO2dVYsBvds&rel=0

Forecast attribute traits

In general, you use forecast attribute traits as predicted values to compare with the actual value of the parent attribute. Typically, such attribute traits represent future PI points but that is not a requirement. Client applications can very easily relate these attributes and trend them together.

Forecast attribute traits are child attributes of the attribute they describe and have the same value type and UOM as the parent. The only exception is when the value of a forecast attribute trait is obtained from a data reference, in which case the source UOM of a defined data reference can be different. You can also create forecast attribute traits for parent attributes that are of any value type. Note that you can have multiple forecast attribute traits defined for a parent attribute.

Location attribute traits

You use location attributes to define longitude, latitude, and altitude information for an asset. You can use this information to identify the location of the asset on a map.

You can create location attribute traits on an element, an event frame, a model, or on their child attributes. However, you can create only one of each attribute trait per element, event frame, or model. The UOM for the latitude and longitude attribute traits should be degree (the default) or at least an angle, whereas the altitude attribute trait can be meter (the default) or another UOM in the Length class. The value type for all location attribute traits must be numeric (double).

Reason attribute traits

You use reason attribute traits on event frames and transfers to enable users to select a reason code for excursions, downtime, and other events. You can set a single reason attribute trait on an event frame or transfer template or an event frame or transfer instance. The reason attribute trait must be an enumeration set that is previously defined, or a system enumeration set delivered with PI AF.

Health attribute traits

You use health attribute traits on elements and models to enable users to set a numeric health score and a health status (for example, healthy, out of service, in maintenance, warning, or error). The HealthStatus attribute trait uses values from the **Health Status** enumeration set, which is delivered with PI AF. Administrators can modify the **Health Status** enumeration set as required.

Analysis start-trigger attribute traits

When users configure analytics to generate event frames, they can optionally elect to store the name of the start trigger in the value of an attribute (string) and mark that attribute with the analysis start trigger trait. This enables clients like PI Vision to indicate the start trigger that created that particular event frame. For more information, see Event frame generation analyses.

Topics in this section

- Set limit attribute traits
- Set forecast attribute traits
- Set location attribute traits
- Set reason attribute traits
- Set health attribute traits

Set limit attribute traits

Set up to seven limit traits for an attribute template or an element attribute to enable users to define the expected range of a process variable or a measurement.

Procedure

1. Choose from the following actions:

To set a limit trait for an ...	Do this ...
Attribute template	a. In the Library browser, select an element template. b. In the viewer, click the **Attribute Templates** tab.
Element attribute	a. In the Elements browser, select an element. b. In the viewer, click the **Attributes** tab.

2. Select the attribute for which you want to set one or more limit traits.
3. Choose one of the following actions:
   - Right-click and click **Limits**.
   - At the bottom of the configuration area, click the **Limits** link.
4. In the Limits window, select the check box beside the limit trait you want to set.
5. In **Attribute**, choose one of the following actions:

- Leave the default attribute name as is.
- Replace the default attribute name with a revised name of your choice.
- If a child attribute already exists that you want to assign to the limit trait, select it from the drop-down list.

> **Note:**
> Limit attribute traits that have already been defined are displayed in bold typeface.

6. Choose from the following actions:

To set a ...	Do this ...
Value for the limit trait	In **Value**, enter a limit value.
Value for the limit trait from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. **Note:** The source UOM of a data reference can be different from that of the parent attribute. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

7. Repeat steps 4 to 6 to set values for other limit traits.
8. Click **OK** when your list of limit traits for the selected parent attribute is complete.

Set forecast attribute traits

Set one or more forecast traits for an attribute template or an element attribute to enable users and client applications to compare predicted values with actual values and plot a trend.

Procedure

1. Choose from the following actions:

To set a forecast trait for an ...	Do this ...
Attribute template	a. In the Library browser, select an element template. b. In the viewer, click the **Attribute Templates** tab.
Element attribute	a. In the Elements browser, select an element. b. In the viewer, click the **Attributes** tab.

2. Select the attribute for which you want to set one or more forecast traits.
3. Choose one of the following actions:
   - Right-click and click **Forecasts**.
   - At the bottom of the configuration area, click the **Forecasts** link.
4. In the Forecasts window, click the **New Forecast** button or link.

5. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as is.
   - Replace the default attribute name with a revised name of your choice.
   - If a child attribute already exists that you want to assign to the forecast trait, select it from the drop-down list.

   Note:
   Forecast attribute traits that have already been defined are displayed in bold typeface.

6. Choose from the following actions:

To set a ...	Do this ...
Value for the forecast trait	In **Value**, enter a forecast value.
Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. **Note:** The source UOM of a data reference can be different from that of the parent attribute. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

7. You can set as many forecasted values as necessary. Repeat steps 4 to 6.
8. Click **OK** when your list of forecast attributes for the selected parent attribute is complete.

Set location attribute traits

Set one or more location traits on an element or event frame template or on an element or event frame to enable users to identify the location of an asset on a map. You can also set a location for a model.

Before you start

You can only set one set of location attribute traits per element, model, or event frame.

Procedure

1. Choose from the following actions:

To set a location trait on ...	Do this ...
Element or model template	a. In the Library browser, select an element or model template. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.

Event frame template	a. In the Library browser, select an event frame template. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.
Element or model	a. In the Elements browser, select an element or model. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.
Event frame	a. In the Event Frames browser, select an event frame. b. Choose one of the following actions: ▪ Right-click and click **Location**. ▪ On the **General** tab, click the **Location** link.

Note:
You can also configure location traits as a child attribute template to an existing attribute template, or as a child attribute to an existing attribute:

- On a template, right-click an attribute template and click **Location of *Element/Event Frame/Model* Template**.
- On an element, event frame, or model, right-click an attribute and click **Location of *Element/Event Frame/Model***.

2. In the Location window, select the check box beside the trait you want to apply, or select the check box in the column header to select all traits.
3. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as is. The default attribute name displays the path relative to the element.
   - Replace the default attribute name with a name of your choice. Ensure that the full path to the element is retained.
   - If a child attribute already exists that you want to assign to the location trait, select it from the list.

Note:
Location attribute traits that have already been defined are displayed in bold typeface.

4. Choose from the following actions:

To set a ...	Do this ...
Value for the location trait	In **Value**, enter a location value.

Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when your list of location attributes for the selected element is complete.

Set reason attribute traits

Use reason attribute traits on event frames and transfers to enable users to select a reason code for excursions, downtime, and other events.

Before you start

You can only set one reason attribute trait per event frame or transfer.

Procedure

1. To designate an attribute as a reason attribute trait for an event frame or transfer, choose from the following actions:

To set a reason attribute trait for ...	Do this ...
Event frame template	a. In the Library browser, select an event frame template. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Transfer template	a. In the Library browser, select a transfer template. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Event frame	a. In the Event Frames browser, select an event frame. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.
Transfer	a. In the Event Frames browser, select a transfer. b. Choose one of the following actions: ▪ Right-click and click **Reason**. ▪ On the **General** tab, click the **Reason** link.

Attribute template for an event frame template	a. In the Library browser, select an event frame template. b. Click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute template and click **Reason for Event Frame Template**. ▪ Click **New Attribute Template**, right-click and click **Reason for Event Frame Template**.
Attribute template for a transfer template	a. In the Library browser, select a transfer template. b. Click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute template and click **Reason for Transfer Template**. ▪ Click **New Attribute Template**, right-click and click **Reason for Transfer Template**.
Event frame or transfer attribute	If the **Allow Extensions** check box has been selected for an event frame or a transfer template, and no attribute template has previously been designated as a reason attribute trait, you can either create a new attribute and establish it as the reason attribute trait, or choose an existing attribute. a. In the Event Frames or Transfer browser, select an event frame or transfer. b. Click the **Attributes** tab. c. Choose one of the following actions: ▪ Right-click an existing attribute and click **Reason for Event Frame** or **Reason for Transfer**. ▪ Right-click below the attribute table and click **Reason for Event Frame** or **Reason for Transfer**.

2. In the Reason window, select the check box beside the Reason trait you want to apply.
3. In **Attribute**, choose one of the following actions:
   - Leave the default attribute name as **Reason**.
   - If an attribute already exists that you want to designate as the reason trait, select it from the list.

   Note:
   Reason traits that have already been defined are displayed in bold typeface.
4. Choose from the following actions:

To set a ...	Do this ...
Default value for the reason trait	a. In **Value Type**, select **Enumeration Value** > **Enumeration Sets** and select an enumeration set to use for the reason trait. The default is the built-in **System** enumeration set. b. In **Default Value**, select the default reason trait value from the selected enumeration set.
Default value from a data reference **Note:** The data reference must be to a digital states tag.	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when the assignment of a default reason trait for the selected event frame or transfer is complete.

Set health attribute traits

Set health attribute traits for an element template or for an element to enable users to establish the health of an asset. You can also set health attribute traits for a model template or model.

Before you start

You can only set one set of health attribute traits per element or model.

Procedure

1. Choose from the following actions:

To set a health trait for ...	Do this ...
Element or model template	a. In the Library browser, select an element or model template. b. Choose one of the following actions: ▪ Right-click and click **Health**. ▪ On the **General** tab, click the **Health** link.
Element or model	a. In the Elements browser, select an element or model. b. Choose one of the following actions: ▪ Right-click and click **Health**. ▪ On the **General** tab, click the **Health** link.

Attribute template	a. In the Library browser, select an element or model template. b. In the viewer, click the **Attribute Templates** tab. c. Choose one of the following actions: ▪ Right-click under the grid and click **Health of Element Template**. ▪ Click **New Attribute Template** and from the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.
Attribute	a. In the Elements browser, select an element or model. b. In the viewer, click the **Attributes** tab. c. Choose one of the following actions: ▪ Right-click under the grid and click **Health of Element**. ▪ Click **New Attribute** and from the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.

Note:
You can also configure health traits as a child attribute template to an existing attribute template, or as a child attribute to an existing attribute by right-clicking and clicking **Properties**. From the **Properties** list, select **Health** > **HealthScore** or **Health** > **HealthStatus**. Then assign values, as described in step 4.

2. In the Health window, select the check box beside the trait you want to apply, or select the check box in the column header to select all traits.
3. In the **Attribute** column of the Health window, choose one of the following actions:
   - Leave the default attribute name as is. The default attribute name displays the path relative to the element.
   - Replace the default attribute name with a name of your choice. Ensure that the full path to the element is retained.
   - If a child attribute already exists that you want to assign to the health trait, select it from the list.

   Note:
   Health attribute traits that have already been defined are displayed in bold typeface.
4. Choose from the following actions:

To set a ...	Do this ...
Value for the HealthScore trait	In **Default Value** (for a template) or **Value** (for an element) , enter a numeric value.

Value for the HealthStatus trait	In **Default Value**, select a Health Status enumeration value: ◦ **Healthy** ◦ **Out of Service** ◦ **In Maintenance** ◦ **Warning** ◦ **Error** ◦ Other Health Status enumeration set values added by an administrator
Value from a data reference	a. In **Data Reference**, select a data reference type. b. In **Settings**, click ... to configure the data reference. c. Click **OK** to close the data reference window. The result of the evaluated data reference is displayed.

5. Click **OK** when your list of health attributes for the selected element (or model) is complete.

Control the display of attribute and attribute template values

Prior to PI AF 2018, PI System Explorer rendered numeric data types in displays with high levels of precision. For example, the Single and Double data types have a precision of seven and fifteen respectively. Such high levels of precision are not always useful in real-time situations, or for PI AF client products: for example, a temperature gauge may only display a precision of three digits, so process engineers are unlikely to require a greater degree of precision for mathematical calculations.

Beginning with PI AF 2018, the **Use DisplayDigits for Attribute and Attribute Template values** system option is enabled and ensures that the default setting of the DisplayDigits property is used to display attribute values: -5, or a total of five digits overall. This is the same default setting used for PI points on PI Data Archive.

When creating attribute templates, you can now control the number of digits you want to see for individual attributes in the **Display Digits** field:

- You can enter a negative number up to -20 to indicate the total number of digits to display.
- You can enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

When you are editing an attribute value, the actual number is displayed rather than the truncated number established by the **Display Digits** field.

For attributes that are not based on a template, the precision of PI point data references is obtained from the PI point itself.

Note:
On upgrades, existing attributes configured with a PI Point data reference are not updated to obtain their DisplayDigits value from the PI point: they are set to -5.

Create attribute templates

Use attribute templates to create a standard set of attributes that are associated with element, event frame, model, and transfer templates.

Procedure

1. In the Library browser, select a template.
2. To create a new attribute template, choose one of the following actions:
   - On the toolbar, click **New Attribute Template**.
   - Right-click the template in the browser and click **New** > **New Attribute Template**.
3. In the **Name** and **Description** fields, enter a unique name and a description for the attribute template.

   Note:
   When you need to search for an attribute, you can search on up to 440 characters of a description.

4. Set the configuration fields, as needed.

Field	Description
Properties	Select one or more of the following properties, as needed. ◦ **Configuration Item** ◦ **Excluded** ◦ **Hidden** ◦ **Indexed** ◦ **Manual Data Entry** For more information on these properties, see Attribute properties. You can set up the current attribute template to be a location attribute trait for a parent object. Select one of: ◦ **Location** > **Longitude** ◦ **Location** > **Latitude** ◦ **Location** > **Altitude** When the parent object is an event frame or transfer, you can set up the current attribute template to be a reason attribute trait. For more information, see Set reason attribute traits. When the parent object is an element or model, you can set up the current attribute template to be an asset health attribute trait. For more information, see Set health attribute traits. **Note:** When you configure a child attribute template, you can also select limit and forecast attribute traits, depending on the value type of the parent attribute template. For more information, see Attribute traits.
Categories	Click and select one or more previously defined categories from the Categories window. You can also create a new category. For more information, see Categorization of objects.

Default UOM	Select a UOM from the list. For more information on UOMs, see Units of measure in PI AF. **Note:** Although a user can change the UOM that is displayed for an attribute in PI System Explorer, the UOM that is defined in the template does not change.

5. Configure the type of value that the attribute holds.
   - For attributes with constant values, set the attribute **Value Type**. Constant attribute values can be numbers, strings, files, date-times, Boolean, URLs, arrays, and more. For more information, see Define constant values for attribute templates.

     Note:
     Do not set a default UOM for integer value types if searches by client applications, such as PI Vision, will be performed on the value or a UOM conversion is involved. For example, an integer value type with a default UOM is not compatible with the UOM groups feature introduced in PI AF 2017 R2. Use the **Single** or **Double** value type instead.

     Note:
     While allowed, the **Anything** value type is not recommended. When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, formulas, and analytics cannot function with attributes that have no value type.

   - For PI point values, PI point arrays, formulas, and table data (including data from relational databases), click **Settings** to configure a data reference. For more information, see Configuration of data references.
6. Optional. Enter a default value for the attribute in the **Default Value** field.
7. Optional. When the **Use DisplayDigits for Attribute and Attribute Template values** system option is selected, you can override the default DisplayDigits property value of -5 and specify the number of digits to display in the **Display Digits** field. Only numeric values in the **Value** field are affected.
   - Enter a negative number up to -20 to indicate the total number of digits to display.
   - Enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

   For more information, see Control the display of attribute and attribute template values and PI System Explorer customization options.
8. To save your work, press CTRL-S or click **Check In**.

After you finish

When you create an attribute based on the template, you need to set the value:

- For constant values, add the value to the attribute directly.
- For data references, you need to create the instance of the data reference. To do this, right-click on the element in the browser and choose **Create or Update Data Reference**. For more information, see Attribute-value updates from PI point data references.

> **Note:**
> The way you configure an attribute can sometimes restrict how the data can be viewed in client applications, such as PI ProcessBook or PI Vision. See Restrictions on viewing time series data for more information.

PI AF data types

The following table contains definitions of the data types in use in PI AF.

Data type	PI Data Archive equivalent	Description	Range
`Boolean`	None	A data type that tracks true or false conditions	`True` or `False` A `Boolean` attribute can be stored into a PI point of type `String` and any of the `Integer` tags, but `Digital` is the preferred representation.
`Byte`	None	An 8-bit unsigned data type	0 to 255 These values can be represented in a larger PI data type such as `Int16`. However, if PI Data Archive contains data outside the byte range, an error is returned when retrieved through PI AF.
`DateTime`	`Timestamp`	Date and time stored internally in 64 bits	• If stored in PI Data Archive: 1 Jan 1970 to 19 Jan 2038, with a precision of 15 microseconds • If not stored in PI Data Archive: 1 Jan 0001 to 31 Dec 9999, with a precision of 100 nanoseconds
`Double`	`Float64`	A 64-bit floating point number	±5.0e−324 to ±1.7e308, 15 digits of precision
`GUID`	None	A unique 128-bit data type used for the global identification of objects	00000000-0000-0000-0000-000000000000 to ffffffff-ffff-ffff-ffff-ffffffffffff The above values can be represented as a `String` PI data type.
`Int16`	`Int16`	A 16-bit signed integer	-32,768 to 32,767
`Int32`	`Int32`	A 32-bit signed integer	-2,147,483,648 to 2,147,483,647
`Int64`	None	A 64-bit signed integer	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 There is no equivalent of an `Int64` in PI Data Archive. For non-numerical representations (such as an ID number), a string may be used. For numerical representations, `Float64` provides some equivalence but will be subject to rounding issues with very large (positive or negative) numbers.
`Single`	`Float32`	A 32-bit floating point number	±1.5e−45 to ±3.4e38, 7 digits of precision

Data type	PI Data Archive equivalent	Description	Range
`String`	`String`	A sequence of multiple characters	PI Data Archive strings limited to 976 characters, and are not unicode

Define constant values for attribute templates

You can assign constant values directly on an attribute template. Although you cannot set actual attribute values in attribute templates, you can define default values.

Procedure

1. In the viewer, use the **Value Type** field to select the data type of the attribute. There are four groups of value types:
   - **Basic Types**

     A data type supported by PI AF, as described in PI AF data types.
   - **Array Types**

     An array contains any of the Basic types as elements.
   - **Enumeration Sets**

     An enumeration set is a list of user-defined constant values. You typically use an enumeration set to define a list of predefined values for an attribute template. When you configure attributes based on that template, you can select a value from the list. Use the **Enumeration Sets** option to select any of the enumeration sets that have already been created in your PI AF database.
   - **Objects**

     An object can be another PI AF attribute, element, or an operating system file.
2. Click in the **Default Value** field.

   The format of the **Default Value** field changes according to the **Value Type**.
3. Configure the value for the attribute, as described in the appropriate procedure for the attribute data type.

Topics in this section

- Configure values for Basic data types
- Configure values for Array data types
- Configure Enumeration Set values
- Configure values for Object data types

Configure values for Basic data types

After you have set **Value Type** to one of the Basic data types, you can define the attribute's value.

Procedure

1. Click in the **Default Value** field. The format of the field changes according to the value type. See PI AF data types.
2. Enter the attribute's value in the **Default Value** field. PI System Explorer resolves the data you enter and when you check in your changes, it sends the resulting value to PI AF. For example, for **DateTime**, it sends the resulting date and time.

Examples

If the value type is **DateTime**, you can type the time in any string format that is supported by PI AF (including PI Time formats) or the .NET DateTime object. Some examples of valid entries:

```
*-5d
May 12, 2009
07 06 2010  10:00:00 AM
09 14 2008  14:00:00
```

To create an attribute with a link as a value, select the value type **String** and enter the URL as the attribute value. Strings that are recognized as absolute URL paths will be displayed as a link. For example, strings starting with `http://`, `ftp://`, `file://` and `www.` are recognized as links, whereas strings starting with `C:\`, and `abc.com` are not. In PI System Explorer, links appear underlined and have an associated tooltip.

Configure values for Array data types

After you have set **Value Type** to one of the Array data types, you can define the attribute's value.

Procedure

1. Click in the **Default Value** field.
2. Click ... beside the **Default Value** field.
3. In the Array window, define your array. The **Default Value** field is set to the appropriate format for the data type you selected for your array elements. See PI AF data types.
4. Click **OK**. PI System Explorer resolves the data you have entered and when you check in your changes, it sends the resulting array values to PI AF.

Configure Enumeration Set values

When you select an enumeration set for the **Value Type**, you can pick the value for an attribute from that set's predefined list of constant values. For more information on enumeration sets, see Enumeration sets.

Procedure

1. Click in the **Default Value** field. The **Default Value** field becomes a drop-down with values from the selected enumeration set.
2. Select a value from the list of predefined constants. When you check in your changes, PI System Explorer sends the resulting value to PI AF.

Configure values for Object data types

After you have set **Value Type** to one of the Object data types, you then define the attribute's value.

Procedure

1. Click in the **Default Value** field. The action button beside the field changes according to the selected value type.

Value Type	Action
<Anything>	While allowed, the Anything value type is not recommended. **Note:** When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, Formulas, and Analytics cannot function with attributes that have no value type.
Attribute	Click ... to open the Attribute Search window. Enter search criteria, as described in Search for attributes on elements.
Element	Click ... to open the Element Browser window, where you can locate and select an element.
File	Click ▸ and select **Upload**. Locate and select a file in the Open window. For acceptable types of attachment files, see `/FileExtensions` in AFDiag utility parameters. **Caution:** Before you upload a file, run an anti-virus check and ensure that the file size does not exceed your storage space. Some file types can pose a security risk, and a warning message will be displayed, asking if you are sure you want to continue. The list of file types that PI System Explorer considers unsafe can change. As a guideline, look at Microsoft's list of file types blocked by Outlook.

2. Click **OK**. When you check in your changes, PI System Explorer sends the resulting value to PI AF.

Add attribute extensions to template-based elements

In some situations, you might want to add another attribute to an existing element. If the element is based on a template that allows extensions, you can define additional attributes in the element itself independently of the template.

Before you start

The **Allow Extensions** check box on the **General** tab for the element template upon which the attribute is based must be checked.

Procedure

1. In the Elements browser, select the element to which an attribute will be added.
2. Choose from the following actions.
   - In the toolbar, click **New Attribute**.
   - Right-click the element in the browser and click **New** > **New Attribute**.
   - Select the **Attributes** tab in the viewer, right-click and click **New Attribute** from the context menu.
3. In the **Name** and **Description** fields, enter a unique name and a description for the attribute template.

   Note:
   When you need to search for an attribute, you can search on up to 440 characters of a description.

4. Configure the type of value that the attribute holds:
   - For attributes with constant values, set the attribute **Value Type**. This setting determines what you can enter in the **Value** field.

     Constant attribute values can be numbers, strings, files, date-times, Boolean, URLs, arrays, and more.

     Note:
     While allowed, the **Anything** value type is not recommended. When the value type is unspecified, many client applications may have difficulty working with that attribute. For example, PI OLEDB Enterprise, PI ProcessBook, and PI Vision would all have reduced capability with such attributes, such as the inability to trend a value numerically. Furthermore, standard PI AF features such as automatic UOM conversion, formulas, and analytics cannot function with attributes that have no value type.

     Optional. Enter or modify the default value for the attribute in the **Value** field.
   - For formulas, PI point values, PI point arrays, and table data (including data from relational databases) click **Settings** to configure a data reference.

     See Configuration of data references for information about configuring different types of data reference.
5. Set other configuration fields, as needed.

Field	Description
Properties	Select one or more of the following properties, as needed. ◦ **Configuration Item** ◦ **Excluded** ◦ **Hidden** ◦ **Indexed** ◦ **Manual Data Entry** For more information, see Attribute properties.

Categories	Click and select one or more previously defined categories from the Categories window. You can also create a new category. For more information, see Categorization of objects.
Default UOM	Select a UOM from the drop-down list. For more information, see Units of measure in PI AF. **Note:** You can change the UOM that is displayed for the attribute in PI System Explorer; however, the UOM defined in the template is not changed.

6. To save your work, press CTRL-S or click **Check In**.

Enumeration sets

You typically use enumeration sets to establish predefined values for attribute templates. When you configure element attributes based on those templates, you can then have users select those values from pre-populated lists rather than typing values manually. This helps ensure you have consistent nomenclature throughout your database.

Hierarchical enumeration values

Beginning with PI AF 2017 R2, you can nest enumeration values in a hierarchy. For example, suppose you have an enumeration set of pump manufacturers, with child enumeration values for pump types. Within each pump type you could also create child enumeration values for different models. You can create as many levels as you need.

Value	Name	Description
0	Manufacturer A	
5	Manufacturer A|Air operated diaphr...	
10	Manufacturer A|Booster	
15	Manufacturer A|End suction	
16	Manufacturer A|End suction|Series I	Temp under 80º c or 176º F
17	Manufacturer A|End suction|Series II	Hot water pump temp under 120º c or 248º F
18	Manufacturer A|End suction|Series III	Chemical pump temp under 120º c or 248º F
20	Manufacturer A|Magnetic drive	
25	Manufacturer A|Multistage	
30	Manufacturer A|Self-priming	
35	Manufacturer A|Vertical sump	
40	Manufacturer B	
45	Manufacturer B|Chopper	
50	Manufacturer B|Dry pit	
55	Manufacturer B|Grinder	
60	Manufacturer B|Vertical well	

Sample hierarchy of enumeration values

When you configure the `Pump` attribute template, you can simply select the `Pump Manufacturer` enumeration set as the **Value Type**, and thereby enable users to select predefined pump types and model numbers.

Sample enumeration set in attribute template

Sort order

Beginning with PI AF 2017 R2, wherever predefined values defined by enumeration sets are displayed, users can select **Sort By** > **Sort By Name** to sort those values by name, or **Sort By** > **Sort By Value** to sort by enumeration set value.

Video

For information on how to create enumeration sets, watch this video:

https://www.youtube.com/watch?v=nH7zvsb8kBE&rel=0

Manage enumeration sets

You use enumeration sets in attribute templates whenever you want to be able to select from lists of predefined values when you are defining element attributes.

Procedure

1. In the navigator, click **Library**.
2. Choose from the following actions.

To ...	Do this ...
Create a new enumeration set	a. Choose one of the following actions. ▪ On the toolbar, click **New Enumeration Set**. ▪ In the browser, right-click the **Enumeration Sets** collection and click **New Enumeration Set**. b. In the **Name** field, enter a unique name for the enumeration set. c. Optional. In the **Description** field, enter a description for the enumeration set. d. Optional. To use hexadecimal values, select the **Hexadecimal** check box. e. Optional. If you want to configure access permissions for the new enumeration set that are different from those inherited from the **Enumeration Sets** collection, click the **Security** link. For more information, see Configure security for objects.
Create a new enumeration value	a. Choose one of the following actions. ▪ In the browser, click an existing enumeration set. In the viewer, right-click in any **Value** field and click **New Enumeration Value**. ▪ In the browser, right-click an existing enumeration set and click **New Enumeration Value**. The value is a unique numeric value associated with the enumeration and provides a quicker, less memory-intensive representation of an enumeration's value. b. In the **Value** field of the Enumeration Value Properties window, either enter a unique number manually, or click to increase or lower the value. c. In the **Name** field, enter a unique string that describes the condition or state being represented. This string is used as the displayed value when an enumeration set is selected as the value type for an attribute. d. Optional. In the **Description** field, enter a description for the enumeration value.

Create a new child enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click an existing value to which you want to add a child enumeration value, and click **New Child Enumeration Value**. c. In the Enumeration Value Properties window, verify that the value selected in the **Parent** field is correct. Otherwise, select a different parent from the list. d. In the **Value** field, either enter a unique number manually, or click to increase or lower the value. e. In the **Name** field, enter a unique string that describes the condition or state being represented. This string is used as the displayed value when an enumeration set is selected as the value type for an attribute. f. Optional. In the **Description** field, enter a description for the enumeration value.
Rename an enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click the existing value that you want to rename and click **Properties**. c. In the **Name** field of the Enumeration Value Properties window, revise the name string as needed but ensure it remains unique. If the name is being used as a parent enumeration value, note that the name change will update the parent portion of all child enumeration value strings.
Renumber enumeration values	See Renumber enumeration values.
Delete an enumeration value	a. In the browser, click an existing enumeration set. b. In the viewer, right-click the existing value that you want to delete and click **Delete**. c. Click **Yes** in response to the Delete warning. Note that you will invalidate objects where the value is currently being used.

3. On the toolbar, click **Check In**.

Renumber enumeration values

You can renumber an enumeration set although you will invalidate objects where the value is currently being used.

Procedure

1. Right-click anywhere in a field and select **Renumber Enumeration Values**.

The Change Enumeration Value confirmation window explains that you will invalidate objects already using the existing enumeration set.

2. To continue, click **Yes**.
3. In the **Starting Values** field of the Renumber window, click to increase or lower the value.
4. In the **Increment by** field, click to select an increment value.

   If you select a row before renumbering, renumbering starts at the selected row, with this row getting the starting value and each subsequent row getting the next value of increment. Only the selected row and following rows are changed during a renumber action. If renumbering does not start at the topmost row, a possibility of the generated values being identical to the values above the selected row exists. The following error message is displayed:

   ```
   Attempting to change the enumeration values would overlap previous
   enumeration values prior to the selected row.
   ```

Configuration of data references

You can configure an attribute or attribute template to obtain a value from a specified source. This configuration is called a data reference. Attributes that include them are called data reference attributes. You create a data reference for the following sources:

- **PI point**

  You use the PI point value, or the value of summary, or other operations on the point value.

- **PI point array**

  You use a PI point array data reference to produce a single value from an array of PI points.

- **Formula**

  You create formulas to produce a calculation result that defines the attribute value. The calculation itself can include attributes, as well as data reference attributes.

- **Table lookup**

  You define a table internally, import values from an external (non-PI) table, or link to values in an external table.

- **String builder**

  You use substitution parameters and functions to manipulate values and output a string.

- **URI builder**

  You use attribute values and substitution parameters to create dynamic links for attributes. You can use the attribute in a notification or in a client application, such as PI Vision.

Topics in this section

- Attribute configuration strings in PI System Explorer
- Substitution parameters in data references
- PI point data references
- PI point array data references
- Formula data references
- String Builder data references
- Table lookup data references
- URI Builder data references

Attribute configuration strings in PI System Explorer

An attribute configuration string describes a data reference. The syntax of a configuration string depends on the type of data reference.

The configuration string for String Builder data references can contain substitution parameters. For attribute templates, any configuration string can contain substitution parameters. See List of PI AF substitution parameters.

In PI System Explorer, when you select an attribute with a data reference, the configuration string appears in the text box below the **Settings** button. You edit configuration strings directly in the text box.

The following table contains examples of configuration strings for different types of data references:

Type of data reference	Sample configuration string
Formula	`D=Density;V=Volume;[D*V]`
	`A=Attribute3;[A*3];UOM=cm`
PI Point	`\\MyPIDataArchiveServer\sinusoid`
	`\\192.168.0.255\ChocMilkMeter;TimeMethod=TimeRange; RelativeTime=-1h;TimeRangeMethod=Total;ReadOnly=False`
PI Point Array	`\\MyPIDataArchiveServer\Point.1|Point.2|Point.3`
String Builder	`"%Attribute% value is";'Attribute1';`
Table Lookup	`SELECT Density FROM [Material Specifications] WHERE MaterialID = @Product`
URI Builder	`https://MyDataServer.int:443/Vision/#Displays/215915/Mine-Truck-10-Brake-Temp?Asset=\\%System%\%Database%\%Element %&StartTime=03%2F21%2F2016 09:26:00&EndTime=&Mode=kiosk`

Topics in this section

- Configuration strings for PI point data references
- Path syntax for references to other attributes

Configuration strings for PI point data references

The attribute configuration string for PI point data references must contain the path to the point. The string can also contain parameters that specify the value that the data reference returns. If specified in an attribute template, the string can contain parameters that specify the point to create. In the string, you separate the parameters with semi-colons.

Examples

- Simple reference to a point on a PI Data Archive server called `MyPIDataArchiveServer`:
  ```
  \\MyPIDataArchiveServer\sinusoid
  ```
- Configuration string referencing the same point, but with a time retrieval specification and specified units of measure:
  ```
  \\MyPIDataArchiveServer\sinusoid;TimeMethod=ExactTime;UOM=°C
  ```
- Configuration string referencing the same point, but returning a total of point values over a time range:
  ```
  \\MyPIDataArchiveServer\sinusoid;TimeMethod=NotSupported;
  TimeRangeMethod=Total;RateConversion=day
  ```
- Configuration string from an attribute template, using substitution parameters:
  ```
  \\%Server%\%Element%.%Attribute%
  ```
- Same configuration string, but with tag creation enabled and point configuration specified:
  ```
  \\%Server%\%Element%.%Attribute%;ptclassname=classic;pointtype=Float32;
  engunits=m3/s;location1=1;location2=30;location4=1;location5=1;pointsource=R
  ```

Data reference parameters that specify values to return

Configuration strings for PI point data references can include optional parameters that specify the value that the data reference returns. The following table lists available parameters:

Parameter	Syntax	Example
TimeMethod	TimeMethod=*time_method* where *time_method* is one of: • After • AtOrAfter • Before • AtOrBefore • Automatic • ExactTime • Interpolated • NotSupported • TimeRange • TimeRangeOverride	TimeMethod=Automatic
RelativeTime	RelativeTime=[*] +|- integer *time_unit* where *time_unit* is one of: • y • M • d • h • m • s	RelativeTime=-1h
TimeRangeMethod	TimeRangeMethod=*method_name* where *method_name* is one of • Maximum • Minimum • PopulationStandardDeviation • Range • StandardDeviation • StartTime • Total	TimeRangeMethod=Total
TimeRangeMinPercentGood[1]	TimeRangeMinPercentGood= *percentage*	

Parameter	Syntax	Example
`CalculationBasis`[1]	`CalculationBasis=` *calculation_basis* where *calculation_basis* is one of: • `EventWeighted` • `EventWeightedExcludeEarliestEvent` • `EventWeightedExcludeMostRecentEvent` • `EventWeightedIncludeBothEnds` • `TimeWeighted` • `TimeWeightedContinuous` • `TimeWeightedDiscrete`	
`RateConversion`[2]	`RateConversion=`*uom* where *uom* is a defined unit of measure	`RateConversion=minute`
`CaseMethod`	`CaseMethod=`*case_method* where *case_method* is one of: • `StartTime` • `EndTime` • `TimeRange` • `Automatic`	`CaseMethod=EndTime`
`UOM`	`UOM=`*uom* where *uom* is a defined unit of measure	`UOM=°C`
`ReadOnly`	`ReadOnly=`*boolean* where *boolean* is one of: • `true` • `false`	`ReadOnly=false`

[1] Used when specifying **`TimeRangeMethod`**.

[2] Used only when `TimeRangeMethod=Total`.

Data reference parameters that specify PI point to create

In attribute templates that specify new PI points, configuration strings for PI point data references include parameters that specify the type of point to create. Point creation parameters consist of a point attribute and value. To specify a new PI point, attribute templates must include the **`pointtype`** attribute as a parameter; other attributes are optional.

Examples

- Create a PI point of type `Float64` and use default settings for the rest of the point configuration:
  ```
  \\%Server%\%Element%.%Attribute%;pointtype=Float64
  ```
- Create a PI point that sets the point attributes shown in the following table:

Point attribute	Setting
`ptclassname`	`classic`

Point attribute	Setting
pointtype	Float32
pointsource	R
location4	1
location5	2
span	200
zero	1100

```
\\%Server%\%Element%.%Attribute%;ptclassname=classic;pointtype=Float32;
location4=1;location5=2;pointsource=R;span=200;zero=1100
```

Path syntax for references to other attributes

To reference other attributes, you use paths that locate attributes on the server, database, and elements in which they reside.

- Use an absolute path to specify the actual server, database, and element where an attribute is located. For example:
  ```
  \\MyPIDataArchive\MyAFDatabase\MyElement|Attribute1
  ```
- Use a relative path to identify an attribute based on its name and its place in the hierarchy of elements and attributes. For example:
  ```
  ..\Element2|Attribute1
  ```

Components of a path

A path is comprised of parts, with each part representing an object or list of objects. Each part of the path is typically separated as follows: by a single backslash (\), with the following exceptions:

- PI AF attributes and PI AF attribute templates are preceded by the pipe character (|).
- The PI Data Archive is preceded by two backslashes (\\).

You can specify a PI Data Archive and a database by name or by globally unique identifier (GUID). A GUID is a unique 128-bit number produced by Windows to identify a particular component, application, file, or database entry. GUIDs must be specified within curly brackets ({ and }). For example:

```
\\{5c64c379-c182-4f35-8d30-78d8c2f84502}\{5c64c379-c182-4f35-8d30-78d8c2f84503}
```

If you specify both the name and GUID, separate them with a semicolon (;). The first one specified takes precedence in the search, so in the following example, the GUID takes precedence:

```
 \\{5c64c379-c182-4f35-8d30-78d8c2f84502};MySystem\{5c64c379-
c182-4f35-8d30-78d8c2f84503};MyDatabase
```

Topics in this section

- Path syntax for PI Data Archive computers
- Path syntax for PI AF databases
- Syntax for relative paths
- Path syntax for dynamic objects
- Path syntax for collection types

- Path syntax for filters
- Wildcard characters in Name filters

Path syntax for PI Data Archive computers

To indicate a fully qualified path from the PI Data Archive, start the path with two backslashes (\\) followed by the PI Data Archive.

To indicate the current system, start a relative path with two backslashes followed by a period (\\.). For example, to reference a database called `Database2` in the current system, type:

```
\\.\Database2
```

To specify a collection of PI Data Archives, use the following format:

```
\\PIDataArchives[MyPIDataArchiveName]
```

Path syntax for PI AF databases

The PI AF database follows the PI Data Archive specification.

You can start a relative path with a single backslash (\) to indicate the current database. For example, the following paths reference objects in the current database:

```
\Element2
```

```
\Tables[MyTable]
```

Note:
You can access non-database objects external to the PI System. You must identify the collection, as demonstrated by the following example:

```
\\MySystem\Contacts[JSmith]
```

Syntax for relative paths

Use a double period (..) to indicate the parent object. The following example references `Attribute1` of `Element2` that is in the current database:

```
..\Element2|Attribute1
```

The single period (.) represents the current object. You can use it to create a relative path from the current object.

- When the current object is a PI AF attribute, a single period followed by a backslash (.\) represents the owning PI AF element. For example:
  ```
  .\|Attribute1
  ```
- A single period followed by a vertical bar (.|) references a child PI AF attribute, for example:
  ```
  .|Attribute1|Attribute2
  ```
- When the current object is a PI AF element, a relative path is created from the database, for example:
  ```
  \Element1\Element2|Attribute1
  ```

Path syntax for dynamic objects

Paths to dynamic objects are also supported. To create a PI point array attribute with `Sinusoid` and `Sinusoidu` values, for example:

```
PI Point Array.\\piserver2\sinusoid|sinusoidu
```

If a path is to a PI point, a dynamic attribute is created. To create an attribute with the data reference configured to read values from the PI point `Sinusoid`, for example:

```
\\piserver2\sinusoid
```

Path syntax for collection types

Each parent object has a default collection type. For example, a PI Data Archive has a default collection of databases, and a PI AF database has a default collection of elements.

Use a single period enclosed in square brackets ([.]) to represent the default object of the parent object. The following example refers to `Element1` of the default database in the current system:

```
\\.\Databases[.]\Element1
```

Path syntax for filters

A collection filter starts with the at sign (@) followed by the filter name. Supported filters are: `@Category`, `@Description`, `@Index`, `@Name`, `@ReferenceType`, `@Template`, `@Trait`, `@Type`, and `@UOM`.

You must enclose the filter specification in square brackets ([and]).

You can also specify multiple filters; they are evaluated in the order specified. For example:

```
\\MySystem\MyDatabase\Elements[@Template=Tank][@Category=Tutorial]|
Attributes[@Category=Tutorial]
```

Category filter example

The following example returns the `Volume` attribute from all elements in the category, `Tutorial`, that belong to the database called `MyDatabase`:

```
\\MySystem\Databases[MyDatabase]\Elements[@Category=Tutorial]|Volume
```

Type filter example

The following example returns the attributes of `Element1` that are of `Int32` type:

```
\Element1|Attributes[@Type=System.Int32]
```

Unit of measure filter example

The following example returns the attributes of `Element1` that have meters as their unit of measure:

```
\Element1|Attributes[@UOM=meter]
```

Index filter example

Use the index filter `[@Index=`*`int`*`]` or `[`*`int`*`]` to specify the position of the matched object to return (the first item is at index position 1). The index filter must be the last filter specified.

The following example returns the third database in the collection of PI AF databases in the current system:

```
\\Systems[MySystem]\Databases[@Index=3]
```

Beginning with PI AF 2017, you can use a negative number for an index from the end of the collection (for example -1 indicates the last item, -2 indicates the penultimate item).

```
\Element#1\Elements[@Name=Tank*][@Index=-3]
```

The index filter name is optional if another filter is specified before the index filter. For example:

```
\Element#1\Elements[@Name=Tank*][3]
```

Wildcard characters in Name filters

You can place a wildcard asterisk (*) character in the name of any object to match zero or more characters.

> **Note:**
> To match a literal asterisk, use a preceding backslash (*).

Examples

```
\\MySystem\MyDatabase\[@Name=E*]
```

```
\\MySystem\MyDatabase\Elements[@Name=E*][@Index=3]
```

Substitution parameters in data references

Substitution parameters are variables that you place in attribute templates for data references such as PI point, String Builder, and URI Builder. PI AF resolves the substitution parameters when it creates the data reference for an attribute based on that template. For example, you can use substitution parameters:

- To configure a PI point data reference template to use names for tags based on element names built from that template.
- To use the value of a different attribute in the configuration of a PI point property value.

Videos

For information on how to use substitution parameters in templates, watch this video:

https://www.youtube.com/watch?v=8jpOjDRwQf0&rel=0

For information on how to use substitution parameters as you create PI points, watch this video:

https://www.youtube.com/watch?v=pTo9nd_L4vI&rel=0

Topics in this section

- Symbols used in substitution parameters
- Name substitutions
- References to attribute values
- List of PI AF substitution parameters

- Format strings for time substitution parameters
- Find default PI Data Archive servers

Symbols used in substitution parameters

Use the symbols in the following table to build a substitution parameter.

Symbol	Description	Examples
`%...%`	Considers the expression as a substitution parameter.	`%Element%` `%Attribute%`
`.`	Current element or attribute. Use `.\` to navigate down from current element. Use `.|` to navigate to child attributes of the current attribute.	`%.\ChildElement|Attribute%`
`..`	Navigates a level up.	`%..\..\Element%` `%..|Attribute%`
`\`	Separates components of a path, except attributes.	`%..\Element%`
`|`	Separates attributes in a path.	`%..|Attribute%`

`@`	References the value of the object instead of its name.
	Note: Only PI point data references use attribute value substitution syntax. Other data references, such as formula, table lookup, and String Builder have a simpler syntax for referencing attribute values.
		• Attribute value at same level as attribute:
		`%@Attribute%`
		• Attribute value at root level of same element:
		`%@|Attribute%`
		• Attribute value at parent attribute level:
		`%@..|Attribute%`
		• Attribute value at parent element level:
		`%@..\\|Attribute%`
		• Attribute value of child attribute of same element:
		`%@.|Attribute%`
		• Attribute value of child element attribute:
		`%@.\ChildElement\ChildOfChild|Attribute%`
		• Attribute value of primary element of event frame:
		`%@.\Elements[.]|Attribute%`

Name substitutions

When you use a name substitution, PI AF performs a direct substitution of the substitution parameter for whatever that particular parameter represents.

The table in List of PI AF substitution parameters lists the available substitution parameters and what they represent. For example, `%Element%` is a substitution parameter that represents the element name. After you create an element based on that template, you tell PI AF to create the data reference (Attribute-value updates from PI point data references). When PI AF creates

the reference, it substitutes the current element name wherever the configuration says `%Element%`.

Suppose you have a data reference template that references the PI point name:

```
%Element%_TT
```

You create an element named Tank1 that is based on that template. The attribute data reference points to a PI point named:

```
Tank1_TT
```

References to attribute values

When configuring a PI point data reference attribute, you can use substitution parameters to reference the value of another attribute. You can use attribute value references to specify a value in the **Relative Time** field or in the point attribute configuration for automatically-generated PI points. The syntax is:

```
%@AttributeName%
```

where *AttributeName* is the name of the attribute. The @ indicates to substitute the value of the indicated attribute, rather than its name. To reference an attribute that is not a sibling of the current attribute, use the syntax described in Indirect PI point references to define the path to the desired attribute.

PI AF does not update the attribute value over time. When a substitution is used in the tag name, the value is resolved only at the time the data reference is loaded. If a substitution parameter is used to define point attributes, PI AF uses the value of the attribute at the exact time that you created or updated the data reference (see Attribute-value updates from PI point data references for more information). This value is a constant. PI AF does not evaluate that attribute value again, unless you update the data reference.

Note:
The exception to this rule is a value substitution for the **Relative Time**, which is evaluated on every value call to the PI point data reference (see Create ranges of configurable durations).

List of PI AF substitution parameters

PI AF interprets numerous substitution parameters.

Name substitution parameters

The following table lists the name substitution parameters that PI AF supports.

Parameter name	Substitution
%@*path*%	The value of the object's attribute or attribute template, represented by the path.
%Analysis%	Name of analysis, if obtainable from the context.
%Attribute%	Name of the object's attribute or attribute template.
%|Attribute%	Name of the root attribute or attribute template that holds this data reference.
%..|Attribute%	Name of the parent attribute or attribute template that holds this data reference.

Parameter name	Substitution
%Database%	Name of the PI AF database in which the attribute resides.
%Destination%	Name of the destination element for the transfer in which the attribute resides.
%Element%	Name of the element in which the attribute resides. For event frames, this refers to the name of the primary-referenced element.
%\Element%	Name of the root element in which the attribute resides.
%..\Element%	Name of the parent element of the element in which the attribute resides. To retrieve further ancestors, use the `..\` notation, such as `%..\..\Element%`.
%EventFrame%	Name of the event frame in which the attribute resides.
%\EventFrame%	Name of the root event frame in which the attribute resides.
%..\EventFrame%	Name of the parent event frame of the event frame in which the attribute resides. To retrieve further ancestors, use the `..\` notation, such as `%..\..\EventFrame%`.
%Model%	Name of the model, if obtainable from the context.
%Name:*path***%**	Name of the object represented by the path to the object. If the path is not specified, null is returned since the name of the current object is being referenced.
%Server%	Name of the default PI Data Archive for the computer on which you create the attribute. When creating the attribute in PI System Explorer, this is the default PI Data Archive for the computer on which PI System Explorer is running. **Note:** The **%Server%** parameter does not resolve to the computer on which the PI AF database resides. The **%Server%** parameter can resolve to a different PI Data Archive depending on the default in PI AF Client.
%Source%	Name of the source element for the transfer in which the attribute resides.
%System%	Name of the PI AF server or collective where the attribute resides.
%Template%	Name of the template on which the element or event frame is based. For example, if you created element `Valve101` from a template called `Valve`, then the substitute text would be `Valve`.
%\Template%	Name of the root template on which the element or event frame is based.
%..\Template%	Name of the parent template on which the element or event frame is based. To retrieve further ancestors, use the `..\` notation, such as `%..\..\Template%`.
%Transfer%	Name of the transfer in which the attribute resides.

Time substitution parameters

PI AF supports substitution parameters that specify a particular time and time zone, depending on the context. Optionally, you can augment these supported parameters by including a format string that specifies the format of the time string. See Format strings for time substitution parameters.

The following table lists the time substitution parameters that PI AF supports.

Parameter name	Substitution
%Duration%	Time span between the start time and end time, if it can be obtained from the time context. In open event frames, obtains the time span from start time to the current time. The time span uses a different format from other time substitution parameters. • You can use standard formats such as "c" (constant), "g" (general, short), or "G" (general, long), for example: **%Duration:c%**. For more information, see the MSDN article Standard TimeSpan Format Strings (https://msdn.microsoft.com/en-us/library/ee372286). • You can also use a custom time span format, as described in the MSDN article Custom TimeSpan Format Strings (https://msdn.microsoft.com/en-us/library/ee372287). Note that you need to define the symbols that separate days from hours, and so on with a string literal. For example, **%Duration:d\.hh\:mm\:ss%** defines a period (.) as the separator between days and hours, and a colon (:) as the separator between hours, minutes, and seconds.
%EndTime%	Local end time, if obtainable from the time context. For event frame-based objects that do not have an end time yet specified, the result is an empty string.
%StartTime%	Local start time, if obtainable from the time context.
%Time%	Local time, if obtainable from the time context.
%UtcEndTime%	Coordinated universal (UTC) end time, if it can be obtained from the time context. For event frame-based objects that do not have an end time yet specified, the result is an empty string.
%UtcStartTime%	Coordinated universal (UTC) start time, if it can be obtained from the time context.
%UtcTime%	Coordinated universal (UTC) time, if it can be obtained from the time context.

ID substitution parameters

The following table lists the ID substitution parameters that PI AF supports.

Parameter name	Substitution
%AnalysisId%	ID of the analysis with which the attribute is associated.
%AttributeId%	ID of the attribute that holds this data reference.
%|AttributeId%	ID of the root attribute or root attribute template in which the attribute resides.
%..|AttributeId%	ID of the parent attribute or parent attribute template in which the attribute resides. Further ancestors can be retrieved by using the `..|` notation, such as `%..|..|AttributeId%`.
%DatabaseId%	ID of the database in which the attribute resides.
%ElementId%	ID of the element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element.
%\ElementId%	ID of the root element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element of the root event frame in which the attribute resides.

Parameter name	Substitution
%..\ElementId%	ID of the parent element in which the attribute resides. For event frames, this refers to the ID of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementId%`.
%EventFrameId%	ID of the event frame in which the attribute resides.
%\EventFrameId%	ID of the root event frame in which the attribute resides.
%..\EventFrameId%	ID of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFrameId%`.
%Id:*path***%**	ID of the object represented by the path to the object. If the path is not specified, the ID of the current object is used. .
%ModelId%	ID of the model in which the attribute resides.
%SystemId%	ID of the PI AF server in which the attribute resides.
%TransferId%	ID of the transfer in which the attribute resides.

Description substitution parameters

The following table lists the description substitution parameters that PI AF supports.

Parameter name	Substitution
%Description:*path***%**	Description of the attribute represented by the path to the attribute. If the path is not specified, the description of the current attribute is used.
%|Description%	Description of the root attribute or root attribute template in which the attribute resides.
%..|Description%	Description of the parent attribute or parent attribute template in which the attribute resides. Further ancestors can be retrieved by using the `..|` notation, such as `%..|..|Description%`.
%ElementDescription%	Description of the element in which the attribute resides. For event frames, this refers to the description of the primary referenced element of the event frame in which the attribute resides.
%\ElementDescription%	Description of the root element where the attribute resides. For event frames, this refers to the description of the primary referenced element of the root event frame in which the attribute resides.
%..\ElementDescription%	Description of the parent element in which the attribute resides. For event frames, this refers to the description of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementDescription%`.
%EventFrameDescription%	Description of the event frame in which the attribute resides.
%\EventFrameDescription%	Description of the root event frame in which the attribute resides.
%..\EventFrameDescription%	Description of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFrameDescription%`.

Path substitution parameters

Path substitutions cannot generally be used to create references to other attributes, because they contain backslash characters. The Path substitution parameters are most useful with String Builder data references when you construct paths for output into strings. The path that is produced does not include the PI AF server or database.

The following table lists the path substitution parameters that PI AF supports.

Parameter name	Substitution
`%ElementPath%`	Path of the base element, the element of an attribute, or the primary referenced element of an event frame.
`%..\ElementPath%`	Path of the parent element in which the attribute resides. For event frames, this refers to the path of the primary referenced element of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\ElementPath%`.
`%EventFramePath%`	Path of the event frame, or the element of an attribute of an event frame.
`%..\EventFramePath%`	Path of the parent event frame in which the attribute resides. Further ancestors can be retrieved by using the `..\` notation, such as `%..\..\EventFramePath%`.

Environment variable parameter

The following table lists the environment variable parameter that PI AF supports.

Parameter name	Substitution
%*Environment Variable***%**	Value of the matching system-environment variable. For example, **`%COMPUTERNAME%`** is replaced with the name of the computer on which the data reference is executing.

Format strings for time substitution parameters

In time substitution parameters, you can include a format string that specifies the format of the time string.

Standard date and time format

Use any standard format string supported by the `DateTime.ToString` method, described in the MSDN article Standard Date and Time Format Strings (https://msdn.microsoft.com/en-us/library/az4se3k1%28v=vs.110%29.aspx#Sortable). In the specification of the time substitution parameter, separate the substitution and the format string with a colon. For example, `%TIME:d%` specifies the local time in a short-date pattern.

The following table provides an abbreviated description of `DateTime` format strings:

Format specifier	Description	Example
d	Short date pattern	6/15/2009 (en-US) 15/06/2009 (fr-FR) 2009/06/15 (ja-JP)

Format specifier	Description	Example
D	Long date pattern	Monday, June 15, 2009 (en-US) Montag, 15. Juni 2009 (de-DE)
f	Full date/time pattern (short time)	Monday, June 15, 2009 1:45 PM (en-US)
F	Full date/time pattern (long time)	Monday, June 15, 2009 1:45:30 PM (en-US)
g	General date/time pattern (short time)	6/15/2009 1:45 PM (en-US) 15/06/2009 13:45 (es-ES) 2009/6/15 13:45 (zh-CN)
G	General date/time pattern (long time)	6/15/2009 1:45:30 PM (en-US) 15/06/2009 13:45:30 (es-ES) 2009/6/15 13:45:30 (zh-CN)
M, m	Month/day pattern	June 15 (en-US)
O, o	Round-trip date/time pattern	2009-06-15T13:45:30.0000000-07:00
R, r	RFC1123 pattern	Mon, 15 Jun 2009 20:45:30 GMT
s	Sortable date/time pattern	2009-06-15T13:45:30
t	Short time pattern	1:45 PM (en-US)
T	Long time pattern	1:45:30 PM (en-US)
u	Universal sortable date/time pattern	2009-06-15 20:45:30Z
U	Universal full date/time pattern	Monday, June 15, 2009 8:45:30 PM (en-US)
Y, y	Year month pattern	June, 2009 (en-US)
Any other single character	Unknown specifier	

Custom date and time format

You can also construct custom patterns in the date and time format string, using time specifiers described in the MSDN article Custom Date and Time Format Strings (https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx). For example, `%TIME:yyyy/MM/dd HH:mm:ss.ffffff%` produces output similar to 2015/03/31 09:28:03.843512.

Find default PI Data Archive servers

The `%Server%` substitution parameter resolves to the name of the default PI Data Archive server for the computer on which you create the attribute. If you are creating the attribute in PI System Explorer, this is the default PI Data Archive server for the computer on which PI System Explorer is running.

Note:
Because the `%Server%` parameter can resolve to a different PI Data Archive server based on the client, it is not recommended for installations that may have more than one PI Data Archive server available.

Procedure

1. On the computer where PI System Explorer is running, open the PI SDK Utility. With PI Data Archive installation defaults, you would choose **Start All Programs** > **PI System** > **AboutPI-SDK**
2. In the PI SDK Utility window, choose **File** > **Connections**.
3. In the **PI Connection Manager** window, choose **Connections** > **Options**. The Connection Options window opens with the default server listed under **Default Server Settings**.

PI point data references

A PI point data reference is a reference from a PI AF attribute to a PI point. The value of the data reference attribute can be the same as the point value, or it can be the result of a calculation based on point values. You can create a PI point data reference for:

- An element attribute or an element attribute template
- A transfer attribute or a transfer attribute template
- An event frame attribute or an event frame attribute template

A PI point data reference for an attribute template has additional features. When you create a PI point data reference in an attribute template, you can:

- Automatically create tags referenced by attributes based on the template.
- Create a naming scheme for attributes based on the template so that attributes built from it have unique names that conform to a consistent structure.

To reference the PI point, you use a direct or an indirect reference. There are two quick methods to create a direct reference, whereas you need to use the PI Point Data Reference window to create a more complex direct reference or an indirect reference.

Videos

For information on how to set up PI point data references, watch this video:

https://www.youtube.com/watch?v=CRejcRe22SA&rel=0

For information on UOMs in PI point data references, watch this video:

https://www.youtube.com/watch?v=023T5si2Dtw&rel=0

Topics in this section

- Direct PI point references
- Indirect PI point references
- Create direct PI point data references from Tag Search results
- Configure direct or indirect PI point data references
- Unit of measure considerations
- Configuration of retrieval methods for attribute values
- PI point value updates from a data reference
- Attribute-value updates from PI point data references

- Configure automatic creation of PI points
- Edit PI point properties
- Preview substitution parameters in PI point data references

Direct PI point references

In the configuration string, a direct point reference uses two backslashes (\\) to reference a PI point on a PI Data Archive. For example, the following configuration string references the *sinusoid* point on a PI Data Archive called MyPIServer:

```
\\MyPIServer\sinusoid
```

For attribute templates, you can also use substitution parameters in the reference. For example, the following configuration string references the *sinusoid* point on the default PI Data Archive of the PI AF database for the attribute:

```
\\%Server%\sinusoid
```

For more information on substitution parameters, see Substitution parameters in data references.

Indirect PI point references

You can configure a PI point data reference to point at another attribute. The referenced attribute must itself be a PI point data reference. This is called an *indirect reference* to whatever PI point the target attribute references. It makes sense to reference by attribute when you need multiple attributes that each use data from the same PI point.

For example, suppose you have an attribute called Level that references a PI point registering a tank level. You want three child attributes to hold the daily average, minimum, and maximum tank levels. If you configure the three child attributes to reference the Level attribute, you can later change the Level attribute to reference a different point and you do not have to reconfigure child attributes. Used in combination with templates, this significantly reduces the amount of configuration required per element instance.

Relative path

To reference another attribute, the configuration string uses a *relative path*. The relative path identifies a data reference attribute based on its name and its place in the hierarchy of elements and attributes. For the PI point data reference, the path must include an attribute path designation (| or ..) in the configuration string so that it is distinguished from a PI point reference.

The following table shows typical configurations for indirect references:

Object	Sample syntax
Top level attribute of same element	`|topLevelAttribute`
Parent attribute	`..`
Child attribute	`.|childAttribute`
Sibling attribute (when not a top level attribute)	`..|siblingAttribute`
From event frame to primary element	`.\Elements[.]|Attribute`
From event frame to attribute of parent element	`.\Elements[.]\..\\|Attribute`
From event frame to attribute of child element	`.\Elements[.]\ChildElementName|Attribute`
Database path	`\TopLevelElement|myAttribute`
Full path	`\\myServer\myDatabase\myElement|myAttribute`

For further details about relative path syntax, see Path syntax for references to other attributes.

Create direct PI point data references from Tag Search results

You can drag a PI point from Tag Search results in the palette to set the name, description (if the PI point has a descriptor), and type of a PI point data reference. If the engineering units property of the PI point matches an existing unit of measure, including case, the **Default UOM** field is also set.

Procedure

1. To configure a PI point data reference:
   - In the Elements browser, select the desired element.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. Select **View** > **Palette** > **Tag Search**, or press CTRL+SHIFT+8.
4. In the Tag Search window, create a search query to retrieve the PI points you want to use as data references. For more information on searching for PI points, see Search for PI points and Syntax for PI point searches.
5. Click **Search** to retrieve the points that match into the results table.
6. Do one of the following:
   - To create a new attribute, drag a PI point from search results onto the **Attributes** grid. A new attribute configured as a data reference based on the PI point is added to the grid.
   - To configure an existing attribute, select the attribute in the viewer and drag a PI point from search results to the attribute **Settings** field to configure the attribute as a data reference based on the PI point.
7. Click **Check In** to save the data reference.

Configure direct or indirect PI point data references

Use the PI Point Data Reference window to configure PI point data references on element attributes.

Procedure

1. From the **Attributes** tab in the viewer, select an attribute.
2. In the attribute configuration section, select **PI Point** from the **Data Reference** list.
3. Click **Settings**.

   Tip:
   You can click in the **Settings** field and enter a tag name to create a direct PI point data reference in `\\serverName\tagname` format. PI System Explorer retrieves the tag name from the default PI Data Archive. For example, if you enter `sinusoid`, the PI point data reference displays `\\yourDefaultDataArchive\sinusoid`.

4. In the PI Point Data Reference window, choose from the following actions to create the data reference.

To ...	Do this ...
Create a direct PI point data reference	a. Leave the default server, or choose a different PI Data Archive from the **Data server** list. **Note:** If the desired PI Data Archive does not appear in the list, click ... and select a PI Data Archive from the PI Data Archives window. b. In the **Tag name** field, type the point name, or click to search for a point on the selected PI Data Archive. **Tip:** You can enter a substitution parameter instead of the tag name. If you are creating a reference in an attribute template, you can use substitution parameters in both the **Data server** and **Tag name** fields. See List of PI AF substitution parameters.
Create an indirect PI point data reference	a. Choose the **Attribute** option. b. Type a relative path to the attribute, or choose one from the drop-down list of attributes that have PI Point data references. To type in a path, use the syntax described in Indirect PI point references. The attribute that is referenced must also have a PI point data reference.

5. Optional. Specify the units that the referenced PI point uses. See Unit of measure considerations.

> **Note:**
> For time weighted totals on referenced PI Point data reference attributes, ensure that you select a UOM from the **Source Units** list that matches the UOM of the source attribute. Even if the default source units appear to match the UOM of the source attribute, do not select **<Default>** but instead select the appropriate hardcoded UOM from the list.

6. In the **Value retrieval methods** section, specify how the attribute gets its value. For example, the attribute value could be the same as the point value, or it could be a result of a calculation on the point value. See Configuration of retrieval methods for attribute values.
7. Use the **Read Only** check box to specify whether you want PI AF to write the attribute value back to the source point. By default, PI AF does not write data to the referenced PI point (the **Read Only** check box is checked). See PI point value updates from a data reference.

Unit of measure considerations

When you define a PI point data reference you can optionally specify the units that the referenced PI point uses. If possible, PI AF automatically changes the attribute type to a value type that is compatible with the specified units.

However, if the attribute is defined by an attribute template, PI AF cannot change the type. Instead, PI AF attempts to convert the value to the template's value type. If PI AF cannot convert the value, it generates an error.

Configuration of retrieval methods for attribute values

Client applications request attribute values for a specific time or for a time range. For example, in PI ProcessBook, the display can optionally provide a time range context (a time range symbol, such as a trend, must be present on the display to enable reception of a time range). You typically configure the data reference to expect either a time or a time range. The attribute value will then be either:

- The value of the point at a specific time. See Configure value retrieval by time.
- The result of a calculation on the point's values over a time range. See Configure value retrieval by time range. For example, the attribute value could be the average of the point values over an hour.

Videos

For information on how to retrieve aggregate values in PI point data references, watch this video:

https://www.youtube.com/watch?v=GU3kp7W7a6w&rel=0

For information on how to retrieve a single value in a PI point data reference, offset by a time period, watch this video:

https://www.youtube.com/watch?v=4Mv5aZWz2j8&rel=0

Topics in this section

- Relative time
- Configure value retrieval by time

- Configure value retrieval by time range
- Time ranges for event frames and transfers

Relative time

Relative time expressions are some number of a number of days (`d`), hours (`h`), minutes (`m`), or seconds (`s`), specified with either a leading plus sign (+) or a leading minus sign (-). The default starting point for relative time is the current time. Therefore, a time of **-8h** is eight hours before the current time. Fractional times are supported. For example, use `-1.5d` for one and one-half days. These are all valid relative times:

```
+1d
-24h
-3.25m
+24s
```

Relative time expressions can contain only one operator, either + or -. For example, this is *not* supported:

```
-1d+1h
```

Configure value retrieval by time

A client application request for an attribute value includes a time. Specify how to interpret that time to retrieve values for the attribute.

Procedure

1. Select an option from the **By Time** list.

Option	Description
After	Returns the first recorded value after the time requested by the client application.
At or Before	Returns a recorded value at the time requested by the client application. If no value exists at the specified time, returns the previous recorded value.
At or After	Returns a recorded value at the time requested by the client application. If no value exists at the specified time, returns the next recorded value.
Automatic	A continuous point (step attribute = 0) is treated as Interpolated, whereas a discrete point (step attribute = 1) is treated as "At or Before."
Before	Returns the first recorded value before the time requested by the client application.
Exact Time	Returns a recorded value at the time requested by the client application. If no recorded value exists at that time, an error is returned.
Interpolated	Returns an interpolated value for the time requested by the client application. Discrete points (step attribute = 1) carry the previous value forward.

> **Note:**
> Do not select the **Not Supported**, **Time Range**, and **Time Range Override** options. These options are for attribute values based on time range calculations.

2. Leave the **Relative Time** field empty, unless you want to set up a time delay.
3. From the **By Time Range** list, select **End Time**.

   If the client application sends a time range rather than a specific time, PI AF returns an error message as the attribute value.
4. Click **OK**.

Set up a time delay

If an attribute is configured to return the value at a specific time, you can configure a time delay from the time requested. This can be useful when you are creating dead-time delayed attributes.

Procedure

1. Configure the **Value Retrieval** settings, as described in Configure value retrieval by time.
2. In the **Relative Time** field, type the relative time of the delay. Use a PI relative time expression. See Relative time.

   > **Note:**
   > You can use substitution parameters to read the relative time from the value of another attribute. See References to attribute values.
3. Click **OK**.

Configure value retrieval by time range

If you want the attribute value to be the result of a summary calculation over a time range, configure the value retrieval by time range.

Procedure

1. Select one of the following options from the **By Time** list.

Option	Description
Not Supported	If the client application sends a time instead of a time range, PI AF returns an error message as the attribute value.
Time Range	Create a default time range to use if the client application sends a time instead of a time range. If you choose this option, you must type a PI relative time expression in the **Relative Time** field. See Create default time ranges for element attributes for details.
Time Range Override	Specify a time range that always overrides the time range specified by the client application. You can specify a fixed duration for the range (Create ranges of configurable durations) or you can calculate the duration dynamically (Configure dynamic time range calculations).

2. Select one of the following options from the **By Time Range** list.

Option	Description
Average	Returns the average value over the time range.
Count	Returns the event count over the time range, when **Calculation Basis** is set to **Event Weighted**. Returns the sum of event time duration over the time range, when **Calculation Basis** is set to any of the time weighted options.
Delta	Returns the difference in value from the end of the time range to the start of the time range.
End Time	Returns the value at the end of the time range.
Maximum	Returns the maximum value over the time range. **Note:** The timestamp value displays the time that the maximum value occurred.
Minimum	Returns the minimum value over the time range. **Note:** The timestamp value displays the time that the minimum value occurred.
Population Standard Deviation	Returns the population standard deviation over the time range.
Range	Returns the range of values over the time range (Maximum-Minimum)
Standard Deviation	Returns the standard deviation over the time range.
Start Time	Returns the value at the Start Time of the time range. **Note:** If you choose the **Start Time** option, you cannot choose **Not Supported** in the **By Time** list, otherwise the following error message is displayed: `The TimeMethod must be supported to support the TimeRangeMethod of "Start Time".`
Total	Returns a totalization over the time range.

3. If the **Calculation Basis** list is enabled, select one of the following options.

Note:
If you select one of the time weighted options for a totalization, source units convert to a rate. Select a unit of time from the list next to the **Source Units** field.

Option	Description
Event Weighted	Evaluates values with equal weighting for each event. No interpolation is done. At least one event must be within the time range to perform a successful calculation. Two events are required for standard deviation. In handling events at the boundary of the calculation, PI AF uses the following rules: ◦ Events at both boundaries are used when there is only one calculation interval. ◦ Events at start time are included when multiple intervals exist and the intervals are in ascending time order. ◦ Events at the end time are included when multiple intervals exist and the intervals are in descending time order.
Event Weighted Exclude Most Recent Event	Behaves the same as **Event Weighted**, except in the handling of events at the boundary of summary intervals in a multiple intervals calculation. Use this option to prevent events at the intervals boundary from being double count at both intervals. With this option, events at the end time (most recent time) of an interval are not used in that interval.
Event Weighted Exclude Earliest Event	Similar to the **Event Weighted Exclude Most Recent Event** option. Events at the start time (earliest time) of an interval are not used in that interval.
Event Weighted Include Both Ends	Events at both ends of the interval boundaries are included in the event-weighted calculation.
Time Weighted	Weights the values in the calculation by the time over which they apply. Interpolation is based on the **`step`** attribute of the point. Interpolated events are generated at the boundaries if necessary.
Time Weighted Continuous	Applies weighting as in **Time Weighted**, but does all interpolation between values as if they represent continuous data (standard interpolation), regardless of the **`step`** attribute of the point.
Time Weighted Discrete	Applies weighting as in **Time Weighted** but does interpolation between values as if they represent discrete, unrelated values (stair-step plot), regardless of the **`step`** attribute of the point.

4. In the **Min percent good** field, enter a value between 0 and 100. This is a percentage representing the minimal amount of time a value must be good before a summary value is returned. For time weighted calculations, the percentage is based on time. For event weighted calculations, the percent is based on event count.
5. Click **OK**.

Topics in this section

- Create ranges of configurable durations
- Configure dynamic time range calculations
- Create default time ranges for element attributes

Create ranges of configurable durations

To guarantee that time ranges are always of the same length, configure time range overrides.

Procedure

1. From the **By Time** list, select **Time Range Override**.
2. In the **Relative Time** field, type a PI **relative time** expression (Relative time).

   PI AF creates a time range based on the end time supplied by the client application and the offset defined by the relative time.

   > **Note:**
   > You can use substitution parameters to read the relative time from the value of another attribute (References to attribute values). If you do this, PI AF evaluates the referenced attribute value at the time of each data request.

Example

To create a one-hour rolling average, you would use the following settings:

- From the **By Time** list, select **Time Range Override**.
- In the **Relative Time** box, type `-1h`.
- From the **By Time Range** list, select **Average**.

With this configuration, the data reference always computes a 1-hour average, even when the client application specifies a different time range.

Configure dynamic time range calculations

Use a PI AF attribute to calculate the time range based on variable factors. You can use the attribute value to determine the time range dynamically. The attribute values must be valid PI relative times, as described in Relative time.

Procedure

1. Configure an attribute to generate the PI relative times. The attribute might reference an enumeration set containing PI relative times, or it might construct PI relative times based on a calculation.
2. Configure the PI point data reference.
   a. From the **By Time** list, select **Time Range Override**.
   b. In the **Relative Time** field, type: `%@AttributeName%` where **AttributeName** is the name of the attribute that generates the PI relative time values.
   c. From the **By Time Range** list, select a calculation method.

Results

With this configuration, the data reference uses the calculated relative time as a time offset that determines the time range.

See Example variable end time to step through an example.

Example variable end time

This example creates a data reference with a variable end time.

Procedure

1. Create an enumeration set, `MyEnumSet`, that contains these three values:

```
-30m
-1h
-2h
```

> **Note:**
> Values must be valid relative time expressions, as described in Relative time.

2. Create an attribute, called `TimeRangeAttribute`, that references the `MyEnumSet` enumeration set.

   You specify the enumeration set when configuring the attribute. From the **Value Type** list, select **Enumeration Sets** > **MyEnumSet**.

3. Create another attribute, called `DataReferenceAttribute` with a configured PI point data reference:

   a. For the PI point, choose any point that has a numeric value. In this example, we will use the `sinusoid` point on a PI Data Archive named PISRV1.

   b. From the **By Time** list, select **Time Range Override**.

   c. In the **Relative Time** box, type: `%@TimeRangeAttribute%`

   d. From the **By Time Range** list, select **Average**.

Results

The `TimeRangeAttribute` attribute lets you choose the time interval for the calculation.

Create default time ranges for element attributes

You can create a default time range for element attributes. A default time range is a time range that PI AF uses when a client application provides a specific time, rather than a time range.

> **Note:**
> Event frame and transfer attributes treat time ranges differently, as described in Time ranges for event frames and transfers.

Procedure

1. In the PI Point Data Reference (or PI Point Array Data Reference) window, use the following value retrieval settings:
   a. From the **By Time** list, select **Time Range**.
   b. In the **Relative Time** field, type a PI relative time expression, as described in Relative time.

      > **Note:**
      > You can also use substitution parameters to read the relative time from the value of another attribute, as described in References to attribute values.

   PI AF creates a time range based on the time specified by the client application and the offset defined by the relative time.

Example

A totalization meter requires a time range in order to deliver a totalized value. If the client application sends a specific time, you want to create a one-hour time range that ends in the specified time. To do that, you would use these settings:

1. From the **By Time** list, select **Time Range**.
2. In the **Relative Time** field, type:

   ```
   -1h
   ```

3. From the **By Time Range** list, select **Total**.

Time ranges for event frames and transfers

Event frames and transfers, by their nature, have a time range associated with them. When a calling application requests a value from an attribute, the value returned depends on the data reference and its configuration:

- When the calling application supplies no time context, PI AF calculates the attribute value using the time range of the event frame or transfer. For example, if an attribute contains a PI point configured with a time range summary option, the summary will be performed over the time range of the event frame or transfer.

If an attribute contains a PI point configured with only the default time option, PI AF returns the value of the point at the end time of the event frame.

- When the calling application supplies a single timestamp, PI AF returns the value at the specified time, even if the time is outside the time range of the event frame or transfer.
- If the calling application supplies a time range, PI AF uses the intersection of the supplied time range and the time range of the event frame. If there is no intersection, a "No Data" event is returned.

  The shaded area in the following figure represents the intersection between an event frame time range and a requested time range. For attributes that contain a PI point configured with a time range summary option, the summary is performed over this intersection. If the attribute contains a PI point configured with only the default time option, the value at the start time of the intersection is returned.

Intersection between event frame time range and requested time range

PI point value updates from a data reference

An attribute with a PI point data reference can write a value to a PI point. In the PI Point Data Reference configuration window, the **Read only** check box determines whether the attribute can write data to the referenced PI point. By default, the check box is selected (Read only = True), meaning that the PI AF attribute cannot change the value of the PI point.

In rare cases, you might want to write the attribute value back to the point value. For example, if the attribute value is the result of a calculation or analysis, you can update the PI point with the result of that analysis. In this case, you would clear the **Read only** check box to allow the attribute to write its value to the point.

Note:
Applications that use the AF SDK **`UpdateValue`** method, such as the PI Analysis Service, can write data to the PI point regardless of whether the **Read only** check box is selected or not.

Attribute-value updates from PI point data references

When you create an attribute based on a template with a data reference, PI AF does not automatically create the PI point data references associated with that element. To create the data references, you must right-click the element in the Elements (or Event Frames) browser and click **Create or Update Data Reference**. PI AF replaces all substitution parameters with the corresponding values at that moment, and creates and locks in the data reference.

Data reference lock-in

When a data reference is locked in, changes to the attribute template no longer affect instances of the attribute. Substitution parameters are also no longer evaluated. However, the following fields are exceptions:

- An attribute value reference is entered in the **Relative Time** field, as noted in References to attribute values.
- Any point attribute change made in the Tag Creation Settings window.

For example, if a point attribute change is made or a point attribute is defined by substitution parameters, the data reference of derived attributes will still be affected whenever you right-click an element and click **Create or Update Data Reference**.

Data reference reset

After an attribute value has been locked in, you can use the **Reset to Template** feature to reset an element to its template's original properties, as described in Attribute-value reset to original properties.

Attribute indicators for updates of PI point data references

PI System Explorer employs distinctive visual indicators for attributes that support the **Create or Update Data Reference** option. A visual indicator is also displayed when an attribute no longer appears to be configured correctly, such as after a server or a tag name has been changed.

The following visual indicators for the **Create or Update Data Reference** option are available:

Icon	Description
	The attribute is associated with a template that contains point creation rules. Use the **Create or Update PI Point** option to create the tag configuration based on the template. Additionally, the point name and ID is resolved.
	The attribute's data reference configuration differs from what is stored in the PI AF database. This can occur when a point is renamed or when the underlying PI Data Archive is replaced. Use the **Create or Update PI Point** option to update the configuration.
	The attribute is a configuration item. See Configuration Item attribute property for more information. **Note:** The **Create or Update Data Reference** option is not applicable for configuration item attributes.

When you review element attributes on the **Attributes** tab, the first entry in the column at left of a data reference attribute's name indicates whether you can run the **Create or Update Data Reference** option. You initiate the **Create or Update Data Reference** option by right-clicking an element or an attribute and clicking **Create or Update PI Point**.

The **Create or Update Data Reference** operation updates a PI point data reference with any changes that have occurred to the PI point; substitution parameters are resolved, and server and point IDs are updated. If the PI point does not yet exist, and the configuration contains a point creation option, it is created.

Attribute update example

When you see the indicator displayed for an attribute, you can update its configuration by right-clicking and clicking **Create or Update PI Point**.

> **Note:**
> The icon does not determine beforehand if any changes to the tag or its configuration will occur.

Attribute indicator with tooltip

As the tooltip shown above indicates, when you update the configuration, point name substitution rules are resolved and the point ID is stored into the attribute's metadata. This results in faster subsequent initialization.

Configure automatic creation of PI points

In an attribute template, you can configure a PI point data reference to create PI points automatically.

Procedure

1. In the Library browser, select an element template.
2. Click the **Attribute Templates** tab.
3. Add a new attribute template to the element template, as described in Create attribute templates.
4. From the **Data Reference** list, select **PI Point**.
5. Click **Settings**.

6. In the **Tag name** field in the PI Point Data Reference window, set up the PI point names for the generated points. Use substitution parameters to enable PI AF to build point names automatically for the points it generates.
   a. Click ▸ to select name substitution and related attribute values from a context menu. For more information on name substitution values, see List of PI AF substitution parameters.

      Note:
      If the attribute value substitution is a string that contains spaces, you must surround the entire tag name in double quotes. Concatenate the entire string first, then substitute the value into the tag name portion of the string with String Builder. For example:

      ```
      "\\%Server%\%Element%.%ID%.%@ConcatStringName%;pointtype=String"
      ```

   b. Select the **Tag Creation** check box.
   c. Click ···.
   d. In the Tag Creation Settings window, specify a point class and point type in the **Point Class** and **Point Type** fields. For more information on PI point classes and attributes, see "PI point classes and attributes" in Live Library (https://livelibrary.osisoft.com).
   e. Default attribute values are displayed for the selected point class. You can edit the default PI point attributes, or you can click **Import** to import PI point attribute values from an existing point that you locate in the Tag Search window.
7. To preview how the selected substitution parameters resolve, click the **Select example instance** link in the **Preview** section.
8. In the Find Derived Elements for *Element_Template* window, select a sample element.
9. Click **OK**.

   The **ConfigString** and **Value** fields display the resolution of the substitution parameters you selected, as well as the tag creation settings.
10. To save your work, press CTRL-S or click **Check In**.
11. Create a new element based on your element template and click on the **Attributes** tab.
12. Right-click the PI point data reference attribute and click **Create or Update Data Reference**.
13. To save your work, press CTRL-S or click **Check In**.

Results

All the points that the new elements reference are created. This only works because the elements were based on a template and the attribute template for the data reference uses the Tag Creation option.

After you finish

After the PI points are created, it might take some time before values are written into PI Data Archive by the specified interface. Until then, the values are displayed as **Pt. Created**.

Edit PI point properties

Edit a PI point property when you need to provide more information on a PI point than default tag property values. For example, it can be helpful to provide a descriptor for a PI point that is created as output from an analysis.

Procedure

1. In the Elements browser, locate the element that contains the PI point you want to edit.
2. Click the **Attributes** tab and select the attribute with the **PI Point** data reference.
3. Click **Settings**.
4. In the PI Point Data Reference window, click .
5. In the PI Point Properties window, choose from the following actions:

To ...	Do this ...
Edit a property value	a. Click the **Value** field beside a property. b. Enter a revised value.
Restore an edited value to PI Data Archive default	a. Click the **Value** field beside an edited property. b. Click and select **Restore default value**.

6. Click **OK**.

Preview substitution parameters in PI point data references

Preview how substitution parameters resolve as you configure a PI point data reference in an attribute template.

Procedure

1. In the Library browser, select an element template.
2. Click the **Attribute Templates** tab.
3. Add a new attribute template to the element template, as described in Create attribute templates.
4. From the **Data Reference** list, select **PI Point**.
5. Click **Settings**.

   Default substitution values are displayed in the **Tag name** field.
6. Use substitution parameters to enable PI AF to build point names automatically for the points it generates. Click to select name substitution and related attribute values from a context menu. For more information on name substitution values, see List of PI AF substitution parameters.
7. To preview how the selected substitution parameters resolve, click the **Select example instance** link in the **Preview** section.
8. In the Find Derived Elements for *Element_Template* window, select a sample element.
9. Click **OK**.

The **ConfigString** and **Value** fields display the resolution of the substitution parameters you selected, as well as any tag creation settings and value retrieval methods (if selected).

10. Choose one of the following actions:
    - To confirm the PI point data reference configuration, click **OK**.
    - To change configuration settings, click **Cancel** and start over by clicking **Settings**.

PI point array data references

You use the PI point array data reference to link attributes of elements to arrays of PI points for either reading or writing their values. Each tag name specified corresponds to a single entry in the array. Configure the PI point array data reference as you would the PI point data reference, with the exception of the **Tag Names** field in the PI Point Array Data Reference window.

Video

For information on how to set up PI point array data references, watch this video:

https://www.youtube.com/watch?v=ozyS9QFFZgM&rel=0

Create PI point array data references

Create a PI point array data reference to link attributes of elements to arrays of PI points for either reading or writing their values.

Procedure

1. To configure a PI point array data reference:
    - In the Elements browser, select the desired element.
    - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. In the attribute configuration panel, choose **PI Point Array** from the **Data Reference** list.
4. Click the **Settings** button.
5. In the PI Point Data Array Reference window, leave the default server, or choose a different PI Data Archive from the **Data server** list.

    Note:
    If the desired PI Data Archive does not appear in the list, click ... and select a PI Data Archive from the PI Data Archives window.

6. In the **Tag name** field, choose from the following actions:
    - To enter PI points directly, enter point names separated by the pipe symbol (|). For example:

      ```
      CDM158|CDT158|SINUSOID
      ```

    - To search for PI points, click and enter search criteria in the Tag Search window. Select multiple PI points on the selected PI Data Archive.
7. Optional. Specify the units that the referenced PI points use. See Unit of measure considerations.

8. In the **Value retrieval methods** section, specify how the attribute gets its value. For example, the attribute value could be the same as the point values, or it could be a result of a calculation on the point values. See Configuration of retrieval methods for attribute values.
9. Use the **Read Only** check box to specify whether you want PI AF to write the attribute value back to the source points. By default, PI AF does not write data to the referenced PI points (the **Read Only** check box is checked). See PI point value updates from a data reference.

Formula data references

You use formula data references to create custom calculations based on other element attributes. Calculations can be a single formula or a sequence of calculations. Formula data references can have multiple input attributes.

Videos

For information on how to create formula data references, watch this video:

https://www.youtube.com/watch?v=vReXkkMueIo&rel=0

For information on UOMs in formula data references, watch this video:

https://www.youtube.com/watch?v=0A_Uy5_nbCs&rel=0

Topics in this section

- Formula data reference operators
- Formula data reference functions
- Units of measure in formula data references
- Configure formula data references

Formula data reference operators

You can use the following operators in formula data references (in order of precedence):

Operator	Precedence
Parenthesis	9 (calculated first)
Unary Minus	8
^	7
* / mod	6
+ -	5
< > <= >= == <>	4
NOT	3
AND	2
OR	1 (calculated last)

You cannot use the assignment operator = at the beginning of any formula.

Note:
Formula data reference syntax uses == (two equals signs) to indicate equality and = (single equals sign) for assignment. Analytics (and performance equation) syntax uses = to indicate equality and := for assignment.

Compound operators

You can choose **If-Then-Else**, a compound operator with operands used as follows:

```
If expr0 Then expr1 Else expr2
```

where *expr0*, *expr1*, and *expr2* are expressions. If *expr0* is true, the value of *expr1* is returned; otherwise, the value of *expr2* is returned.

Formula example

This sample formula calculates the daily average cost of reagent per tonne processed, as shown in the **Value** field for **Grinding Media** in the following mineral processing illustration:

```
A=.|Addition (Daily Total);B=.|Unit Cost;C=Tonnage (Past 24 Hours);[if C ==0
then digstate("No Data") else A*B/C]
```

Sample formula data reference

Formula data reference functions

PI AF supports the following functions in a formula data reference.

Function	Description
abs(X)	Absolute value of X.
acos(X)	Arc cosine of X.
asin(X)	Arc sine of X.

Function	Description
atan(X)	Arc tangent of X.
badval(x)	Returns 1 if the value has a bad status, has a floating point value of not a number (NaN), or is not a numeric value, otherwise returns 0.
ceiling(X)	Smallest integer not less than X.
cos(X)	Cosine of X.
cosh(X)	Hyperbolic cosine of X.
cot(X)	Cotangent of X.
coth(X)	Hyperbolic cotangent of X.
csc(X)	Cosecant of X.
csch(X)	Hyperbolic cosecant of X.
digstate()	Returns a system digital state value. • If successful, the system digital state is returned as a system enumeration value. For PI AF Client versions older than 2.4, the value is returned as a PI SDK digital state, because the system enumeration value set did not exist. • If unsuccessful, the value is returned as a string. In both cases, the `AFValue.IsGood` flag is set to false, and the attribute is flagged in the Attributes Viewer with the icon.
e()	Value of the natural logarithm base.
exp(X)	Natural logarithm base e raised to power of X.
floor(X)	Largest integer not greater than X.
ln(X)	Natural logarithm of X.
log(X)	Logarithm of X using base 10.
logbase(X,Y)	Logarithm of X using base Y.
max(X,Y)	Greater of X and Y.
min(X,Y)	Lesser of X and Y.
normalrnd(X,Y)	Random number that maps the normal distribution curve. X is the mean and Y is the standard deviation.
pi()	Value of pi.
poisson(X)	Random number that maps the Poisson distribution. X is the mean.
pow(X,Y)	X raised to the power of Y.
rand()	Uniform random number. The values can be between 0 and less than 1.0
rand(X,Y)	Uniform random number. The values can be between X-Y/2 and less than X +Y/2.
remainder(X,Y)	Returns the remainder resulting from the division of X by Y.
round(X)	X rounded to the nearest whole number.
roundfrac(X,Y)	X rounded to the number of fractional digits specified by Y. Y is an integer number.
sec(X)	Secant of X.
sech(X)	Hyperbolic secant of X.
sign(X)	Returns 1 if X is greater than zero and -1 if X is less than zero.
sin(X)	Sine of X.
sinh(X)	Hyperbolic sine of X.

Function	Description
sqrt(X)	Square root of X.
tan(X)	Tangent of X.
tanh(X)	Hyperbolic tangent of X.

Units of measure in formula data references

When working with formulas and units of measure, you need to configure the equation inputs and the output with the correct UOM:

- For the attributes in the equation, use the UOM that the formula expects (NOT the units the attribute is already in).
- For the calculation result, specify the UOM in which you want the result to appear. This must be consistent with the UOMs for your inputs (attribute values).

Example of UOM in a formula data reference

Consider the following formula configuration for converting volume and density into mass:

```
V=Volume;UOM=L;D=Density;UOM=kg/L;[V*D];UOM=kg
```

The units of measure for the inputs and outputs are consistent: (L*kg/L = kg). This formula works on any input attributes or output attributes, regardless of the attribute's unit of measure, as long as they have a unit of measure of the correct class specified.

Configure formula data references

Create formula data references to perform custom calculations.

Before you start

The formula data reference does not support strings or other non-numeric value types.

Procedure

1. Choose one of the following actions.
   - In the Elements browser, select the desired element.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. From the **Data Reference** drop-down list, choose **Formula**.
4. Click the **Settings** button to open the Formula Data Reference configuration window.
5. If your equation requires an attribute value, click to add it as a parameter (Define parameters for formula data references).
6. Select the **Default Values Allowed** check box to enable PI AF to use attribute default values in the calculation. Default values are specified on the attribute template. If you clear the check box, the calculation fails when the data for one or more attributes is not available.

7. Click to begin defining the equation. For more information, see Define equations for formula data references.
8. In the **Unit of Measure** field, specify the unit of measure the result of the formula or calculation sequence will produce, as described in Units of measure in formula data references.
9. In the **Minimum** field, enter an appropriate value for the minimum returned value. If the calculated value is less than this minimum, the data reference returns the minimum value you specify. If there is no minimum value for this calculation, leave the **Minimum** box blank.
10. In the **Maximum** field, enter an appropriate value for the maximum returned value. If the calculated value is greater than this maximum, the data reference returns the maximum value you specify. If there is no maximum value for this calculation, leave the **Maximum** box blank.

    Note:
    Select the **Stepped** check box for the value to be stepped when plotted.

11. Click **Evaluate** to test the data reference. The value returned by the calculation sequence appears in the adjacent box. When the formula data reference is configured under a template attribute, the calculation uses the default values for the template attributes. When the configuration is done under an element attribute the actual data is used for the calculation.
12. Click **OK**.
13. Configure the rest of the attribute settings as you would for any other attribute.

Topics in this section

- Define parameters for formula data references
- Define equations for formula data references

Define parameters for formula data references

When equations require attribute values, add those attributes as parameters in the Formula Configuration window.

Procedure

1. Click next to the **Parameters** field.

   Note:
   You can alternatively click in an empty row and enter a variable (a -z).

2. In the Parameter Configuration window, select a variable from the **Variable** menu. The number of variables available is limited to 26 chars (from A to Z).
3. Select the attribute that the variable represents.
   - Choose the attribute from the **Attribute** menu. All sibling attributes with supported value types are listed.
   - If the attribute that you want is not in the menu, choose **Other**. A new field called **Attribute** appears in the window. Click to pick your attribute from the tree.

Alternatively, type in the path to the attribute of interest. Examples of some attribute paths are listed.

4. Select the desired unit of measure for the attribute. Note that this is not the UOM of the attribute but rather the UOM that your formula requires from this input, as described in Units of measure in formula data references.

   For example, if your input attribute has a UOM of Celsius but your equation requires Fahrenheit, you would select Fahrenheit in this field.

Define equations for formula data references

Define equations for use in formula data references.

Procedure

1. To add a new formula row, click next to the **Equations** field. You can also click in the text area and type in an equation.
2. Type an equation directly or build your equation from a list of available variables, operators, functions, and substitution parameters. Click to choose from the following actions:

To ...	Do this ...
Enter a variable	Click **Variables** and click a letter on the alphabetized list.
Enter an operator	Click **Operators** and click an operator on the list. Valid operators are listed in Formula data reference operators.
Enter a function	Click **Functions** and click a function on the list. Valid functions are listed in Formula data reference functions.
Enter a substitution parameter	Click **Substitution Parameters** and click a time substitution parameter. For elements and transfers, only **`%Time%`** is available, but for event frames you can select from **`%StartTime`**, **`%EndTime%`**, **`%UtcTime%`**, **`%UtcStartTime%`**, **`%UtcEndTime%`**, and **`%Duration%`**. For more information on these parameters, see List of PI AF substitution parameters. Numbers are converted to a double, time spans to a duration in seconds, and timestamps to UTC seconds.

3. To add another row to the formula, click and continue the formula. Repeat as needed until you have completed the formula.
4. To change the formula, choose from the following actions:

To ...	Do this ...
Remove the selected equation from the calculation sequence	Select the row you wish to remove and click .
Remove all equations from the calculation sequence	Click .
Move an equation up or down in the calculation sequence	Select the row to be moved and click or .

String Builder data references

The **String Builder** data reference enables you to apply string manipulation functions, such as concatenation, to an attribute's values, and output a reformatted string. This is useful when you need to obtain a string or numeric value type from other element attributes. String Builder data references do not perform Unit of Measure conversions or calculations. When you use the **String Builder** data reference in a template, value substitution takes place at run time.

Expression syntax

Observe the following rules as you build an expression:

- You can construct an expression on a single line, using semicolons to separate its terms.
- You can also place each term on its own line, which eliminates the semicolons and makes the expression structure more apparent.
- Enclose substitution parameters in double quotes so that their result is treated as a literal string.
- You can include any keyboard character in a double-quoted string, for example, dashes, backslashes, colons, semicolons, and so on. To include the double-quote character in a string, type two consecutive double-quote characters within the string, for example, `"""t""e""s""t"""` produces `"t"e"s"t"`.
- You can include single-quoted attribute paths to display the value of the referenced attribute in the result.
- You should avoid the use of unquoted strings because you may get haphazard results.

Topics in this section

- Attribute references in String Builder
- Function implementation in String Builder
- Create String Builder data references

Attribute references in String Builder

To reference other attributes, you use the syntax in the following table.

Object	Syntax
Sibling attribute	`siblingAttribute`
Top level attribute of same element	`|topLevelAttribute`
Parent attribute	`..`
Parent element attribute	`..\\|myAttribute`
Child attribute	`.|childAttribute`
From event frame to primary element	`.\Elements[.]|myAttribute`
Database relative path	`\TopLevelElement|myAttribute`
Full path	`\\myServer\myDatabase\myElement|myAttribute`

Examples

Suppose you want to retrieve the `Environment` PI point value from the `Cracking Process` parent element:

In the child element `Equipment`, you would use the syntax `..\|myAttribute` and enter `'..\|Environment'` in the String Builder Data Reference window:

A value similar to the following is displayed in the **Value** field: **43.3071403505214**.

Function implementation in String Builder

In String Builder, you can use several text manipulation functions, as well as the **`Format`** function.

Text extraction and case manipulation

You can use functions to manipulate the case of a string and to extract certain sections.

The syntax for text manipulation functions in String Builder is described in the following table:

`Left(string, length)`	Returns a string that contains the specified number of characters (`length`) from the left of the input (`string`). Example: `Left("Temperature",4)` returns `Temp`
`Right(string, length)`	Returns a string that contains the specified number of characters (`length`) from the right of the input (`string`). Example: `Right("GasTemp",4)` returns `Temp`
`Mid(string, start, [length])`	`length` is optional. Returns a sub-string from the specified position (`start`) of the input (`string`). When number of characters (`length`) is included, returns the specified number of characters. Example: `Mid("GasPressure",4,8)` returns `Pressure`

UCase(string)	Converts `string` to uppercase. Example: `UCase("Temperature")` returns `TEMPERATURE`
LCase(string)	Converts `string` to lowercase. Example: `LCase("TEMPERATURE")` returns `temperature`
Trim(string)	Removes blanks on both sides of `string`. Example: `Trim(" Temperature ")` returns `Temperature`
RTrim(string)	Removes trailing blanks from `string`. Example: `RTrim(" Temperature ")` returns `" Temperature"`
LTrim(string)	Removes leading blanks from `string`. Example: `LTrim(" Temperature ")` returns `"Temperature "`
Replace(string, string, string)	In original `string`, replaces specified `string` with new `string`. Example: `Replace("Temperature","Temp","External Temp")` returns `"External Temperature"`

You can also nest text manipulation functions.

- For example, for an attribute named `GasPressure`, you can use the **`Mid`** function in combination with the **`UCase`** function to return the following expression: `GASPR`
  ```
  Mid(UCase("%Attribute%"), 1, 5);
  ```
- Alternatively, for the same attribute, you can use the **`Mid`** function in combination with the **`LCase`** function to return the following expression: `pressure`
  ```
  Mid(LCase("%Attribute%"), 4, 8);
  ```

Note:
Beginning with PI AF 2018, you can nest functions in any position, not the first position only. For example,

```
Replace(Mid("abc",1,3),"b","2");
Replace("abc",Mid("abc",2,1),"2");
Replace("abc","b",Mid("123",2,1));
```

all return the expression: `a2c`

Format function

The **`Format`** function enables you to convert real numbers, integers, and time stamps to a string according to the format and optional culture specification.

Note:
Unlike other string functions, the syntax of **Format(DateTime, ...)** follows C# syntax.

The syntax for **`Function`** implementation in String Builder is described in the following table:

Real numbers	**`Format(real, format)`** **`Format (real, format, culture)`**	Format follows Performance Equation (PE) style syntax, such as `"%0.3f"`, where the number before the decimal indicates the minimum total number of characters to output, pre-padding with blanks, and the number after the decimal indicates the number of digits to display after the decimal point. Culture is optionally specified using an IETF language tag, such as `"en"`, `"en-US"`, `"de"`, which specifies the language and optional culture and regions.
Integer numbers	**`Format(integer, format)`** **`Format(integer, format, culture)`**	Format follows Performance Equation (PE) style syntax, such as `"%4d"`, where the number indicates the minimum total number of characters to output, pre-padding with blanks as necessary. Culture is optionally specified using an Internet Engineering Task Force (IETF) language tag, such as `"en"`, `"en-US"`, `"de"`, which specifies the language and optional culture and regions.
Time stamps	**`Format(datetime, format)`** **`Format(datetime, format, culture)`**	Format follows C# `DateTime.ToString(format)` syntax, and can be either a pre-defined syntax, or a custom syntax. `DateTime` format uses invariant culture settings. To display dates and times for a specific culture, add an Internet Engineering Task Force (IETF) language tag, such as `"fr-FR"` and `"fr-CA"`, as described in the MSDN article Table of Language Culture Names, Codes, and ISO Values Method (https://msdn.microsoft.com/en-ca/library/ee825488(v=cs.20).aspx). For more information on date and time formats, see Format strings for time substitution parameters.

Numeric format example

Suppose you want to format the `Environment` PI point value that you have already retrieved from the `Cracking Process` parent element. In the String Builder Data Reference window, you would select the row that contains the string expression, click ▸ and select **Functions** > **Format(Real, "%3.3f")** to modify the expression:

The **Value** field changes from `43.3071403505214` to `43.307`.

To change the cultural value from the US default to a Spanish culture format, you would select **Functions** > **Format(Real, "%3.3f", "en-US")** and change the `"en-US"` string to `"es"`:

The **Value** field changes from `43.307` to `43,307`.

> **Note:**
> Be sure the data type you specify matches the attribute data type. You would encounter errors, for example, if you specify `"%3.2f"` for an integer type attribute value or `"%3d"` for a floating point attribute value.

Date time format example

Suppose you want to retrieve and format the `In Service Date` value from the `Condenser` child element:

In the parent element `Power and Steam Generation`, you would create an attribute named `Condenser Service Date`. Use the syntax `.\childElement|Attribute`, and enter `'.\Condenser|In Service Date'` in the String Builder Data Reference window:

The `In Service Date` value is retrieved and displayed in the **Value** field: `2/25/2009 12:00:00 AM`.

To change the format from the US default to Universal full date in German culture format, you would alter the expression to read `Format(.\Condenser|In Service Date,"U","de")`. The `Condenser Service Date` attribute value format is converted accordingly:

Create String Builder data references

Create a String Builder data reference when you need to apply string manipulation functions, such as concatenation, to an attribute value and output a reformatted string.

Before you start

When a String Builder data reference is used in a template, value substitution takes place at run time.

Procedure

1. Choose one of the following actions:
   - In the Elements browser, select the desired element.
   - In the Event Frames browser, select the desired event frame.
   - In an element template, select the desired element template in the Library browser.
2. In the **Attribute** or **Attribute Templates** tab, select the attribute.
3. In the attribute configuration panel, choose **String Builder** from the **Data Reference** drop-down list.
4. Click **Settings**.
5. In the String Builder Data Reference window, click .
6. In the highlighted row, press F2 or click and select from the following options:

Option	Description
Literals	A text string between double quotation marks: "Sample"
Attribute Values	A list of other attributes for the selected element or element template. Note that attributes must appear in single quotes.
Related Attribute Values	A list of the following attribute references: ◦ '\\<Server>\<Database>\<Element>|<Attribute>' ◦ '\\.\<Database>\<Element>|<Attribute>' ◦ '\<Root Element>|<Attribute>' ◦ '.\<Child Element>|<Attribute>' ◦ '..\\|<Primary Parent Attribute>' ◦ '..\<Sibling Element>|<Attribute>' You can also select **Search** to enter search criteria in the Attribute Search window.
Substitution Parameters	A list of 20 commonly used substitution parameters. For more information, see Substitution parameter list.

Functions	A list of the following functions: ◦ Left(string, length) ◦ Right(string, length) ◦ Mid(string, length) ◦ Mid(string, start, length) ◦ UCase(string) ◦ LCase(string) ◦ Trim(string) ◦ RTrim(string) ◦ LTrim(string) ◦ Replace(string, string, string) ◦ Format(Real, "%3.3f") ◦ Format(Real "%3.3f", "en-US") ◦ Format(DateTime, "yyyy-MM-dd HH:mm:ss") ◦ Format(DateTime, "yyyy-MM-dd HH:mm:ss", "en-US") For examples of how these functions can be used, see Function implementation in String Builder.

7. Build your expression further. You can either type a semicolon (`;`) and continue in the same row, or click to continue on another row. You can use the following icons to manipulate the expression:

To ...	Do this ...
Append a new string	Click .
Remove a string	Select a row and click .
Remove all strings	Click .
Move a string up a row	Select a row below the top row and click .
Move a string down a row	Select a row above the bottom row and click .

As you build the string expression, you can preview the result of the expression in the **Value** field.

8. When the string expression is complete, click **OK**.

Examples

- You can enter the following substitution parameters in a single row to concatenate a pathname string. For example:

```
"%Database%";"\";"%Element%";"|";"%Attribute%";
```

This data reference would produce output similar to: `DB26\WX12000|pressure`.

- You can build an expression in separate rows to show the duration of an event. For example:

a. Click , press F2 and select **Substitution Parameters** > **"%StartTime%"**.

b. Click and type the characters: `" to "`

c. Click , press F2 and select **Substitution Parameters** > **"%EndTime%"**.

The **Value** field shows a date and time interval:

Table lookup data references

You can configure an attribute to obtain its value from a PI AF table. You can define the PI AF table entirely in PI AF, or you can link to or import from a data source outside the PI System (such as a Microsoft SQL table or a Microsoft Excel worksheet).

Table lookup

The table lookup data reference is intended to be a simple table lookup. It uses Microsoft's .NET **`System.Data.DataTable.Select`** method, and follows the filter expression rules of that method. Note that not all its features are implemented in PI AF. For example, a table lookup data reference does not include functionality such as column functions and parent/child relation referencing.

Note:
The table lookup data reference is not optimized to treat data in an external table as a time series and query for it. For optimal performance, such data should be stored in PI tags on a PI Data Archive.

Table lookup methodology

You begin by creating a table profile in the Library browser, where you establish general settings for the time zone and cache interval for refreshing data, as well as definitions of table columns and connections to data sources outside the PI System if you are linking to or importing external data.

If you are linking to external data sources, consider creating reusable table connection profiles in the Library browser, as described in Create reusable table connections.

Videos

For information on how to create internal tables for table lookup data references, watch this video:

https://www.youtube.com/watch?v=3gQzzNWzd6o&rel=0

For information on how to configure table lookup data references, watch this video:

https://www.youtube.com/watch?v=hoy7QarxF1o&rel=0

Topics in this section

- Create PI AF tables
- Create table column definitions
- Populate tables manually
- Data references from outside the PI System
- Configure table lookup data references

Create PI AF tables

You create a table profile so that attributes can obtain their values from an internal or external table.

Procedure

1. In the Library browser, right-click the **Tables** collection and click **New Table**.
2. In the **General** tab, enter a **Name** and **Description** for the table.
3. Optional. Assign one or more categories to the table. Enter table category names directly, separated by a colon, or click and select them from the Categorize window.
4. Decide how `DateTime` values should be stored in the table. By setting the value for the table overall, you need not specify it every time you create a table lookup data reference.
   a. In the **Time Zone** field, select a time zone from the list.
   b. Select or clear the **Convert To Local** check box. Choose from the following options:
   - To convert `DateTime` values to the local time, select the **Convert To Local** check box.
   - To display `DateTime` values in the time zone selected in the **Time Zone** field, clear the **Convert To Local** check box.

   Note:
   When a client application queries the table, the time zone in which `DateTime` values are displayed depends on whether the **Convert To Local** check box is selected. PI System Explorer uses this setting to determine the precise `DateTimeMode` to use in a Microsoft .NET data table, as described in Conversion settings for time zones.
5. In the **Cache Interval** field, enter the amount of time until the table's cached data is automatically refreshed. From the list, choose whether the value is in seconds, minutes, hours, or days. The default value is zero, which indicates a manual refresh.

> **Note:**
> Automatic refreshing is disabled if the table has changes that have not been saved to the server.

> **Note:**
> With manual refreshes, you refresh table data by right-clicking the table in the Library browser and clicking **Refresh**. Otherwise, table data that is being queried by client applications is refreshed only when the application is restarted. For example, when PI System Explorer is opened or whenever the Microsoft Internet Information Services (IIS) application pool recycles the client (the default is every 29 hours).

6. Check in your work.

> **Note:**
> The **Connection** and **Query** fields are read-only. PI AF populates these fields when you link to or import an external table.

After you finish

Define and populate the table in one of three ways:

- Manually define and populate the table in PI System Explorer. First, create column definitions on the **Define Table** tab, as described in Create table column definitions. Next, enter data on the **Table** tab, as described in Populate tables manually.
- Import a table from outside the PI AF server. See Access to Microsoft Access or Excel data.
- Link to a table outside the PI AF server. See Access to Microsoft Access or Excel data.

Conversion settings for time zones

The precise time conversion calculation depends on:

- The value that you select in the **Time Zone** list. Times are prefixed with either Universal Coordinated Time (UTC), Greenwich Mean Time (GMT), or Time Zone (TZ). The list also includes the None option.
- Whether you selected or cleared the **Convert to Local** check box.
- Whether this is an internal (either imported or defined in PI AF) or external (linked) table.

The following table explains how displayed times are determined for each possible combination. The exact options in the **Time Zone** list are dependent on your operating system.

Time zone	Convert to local	Internal/ external	Behavior
None	✓	Internal	Fields that contain `DateTime` data type are automatically adjusted for time zone differences and return local times on the client.
		External	Fields that contain `DateTime` data type are assumed to be in the time zone of the PI AF server and are adjusted to the client's local time.

Time zone	Convert to local	Internal/ external	Behavior
None	✗	Internal	Fields that contain `DateTime` data type are not translated. Useful for storing date fields where translation is not desired, such as birth date.
		External	Fields that contain `DateTime` data type are not translated. Useful for storing date fields where translation is not desired, such as birth date.
`UTC` or `GMT`	✓	Internal	Fields that contain `DateTime` data type are adjusted to local times on the client.
		External	Fields that contain `DateTime` data type are externally stored as UTC, but adjusted to local time on the client, represented as UTC.
`UTC` or `GMT`	✗	Internal	Fields that contain `DateTime` data type are always represented as UTC.
		External	Fields that contain `DateTime` data type are externally stored as UTC, and left as UTC on the client.
`TZ`	✓	Internal	Fields that contain `DateTime` data type are adjusted for time zone differences and are local time on the client. This combination is not normally needed except when external data is being imported into PI System Explorer.
		External	Fields that contain `DateTime` data type are adjusted for time zone differences, and are adjusted to local time on the client, represented as UTC.
`TZ`	✗	Internal	Fields that contain `DateTime` data type are left in the time zone specified. A table lookup data reference adjusts as appropriate; however, other applications may not.
		External	Fields that contain `DateTime` data type are left in the original time zone specified, represented as UTC. A table lookup data reference adjusts as appropriate; however, other applications may not.

Create table column definitions

To create a table manually, begin by defining the columns in the table.

Procedure

1. Once you have completed the overall table settings on the **General** tab, click the **Define Table** tab.
2. Determine the number of columns in the table and click once for each column to be defined.
3. Click the **Name** cell and replace the default label with the required column name.
4. Click the adjacent **Value Type** cell and select a data type from a list of **Basic Types** or **Array Types**. For a description of data types, see PI AF data types.

> **Tip:**
> If any value type is set to `DateTime` or `String`, OSIsoft recommends that you leave the column time zone value set to **None** and select a time zone value for the table overall in the **Time Zone** field on the **General** tab. That value supersedes time zones set on specific table columns.

5. Optional. Click the **Unit of Measure** cell and choose one of the following actions:
   - Start typing and select a unit of measure from the list of matching UOMs.
   - Click ˅ and select from the full list of UOMs.

> **Tip:**
> OSIsoft recommends that you set a unit of measure for a table column so that you do not need to specify one each time you select the column in a table lookup data reference.

6. Repeat steps 3 to 5 for each column in the table.
7. To modify the table column definition, right-click a row in the grid and select an option. Alternatively, use the following icons.

To ...	Do this ...
Insert a new column	Click .
Move a column up a row	Select a row below the top row and click .
Move a column down a row	Select a row above the bottom row and click .
Remove a column	Select a column and click .

After you finish

When you have completed defining table columns, click the **Table** tab so you can enter row data for each column manually.

Populate tables manually

Once you have completed table column definitions, enter row data for each column.

Procedure

1. Click the **Table** tab. Based on the table columns that you created in the **Define Table** tab, enter the appropriate information in rows for each column.
2. Choose from the following actions:

To ...	Do this ...
Enter a new row	Right-click in the table grid and click **Insert** or **New**.
Copy and paste rows from an Excel spreadsheet	a. In the spreadsheet, copy the rows you want. b. In the table grid, right-click a new row or a range of rows and click **Paste**.

Replace data for specific cells	a. Use standard Windows selection keystrokes (such as SHIFT+<click> and CTRL+<click>) to select contiguous and non-contiguous cells in the table, as described in Select multiple objects in the viewer. b. Right-click and click **Clear Cell(s)**. c. Enter new data in the cleared cells.
Remove a row	Right-click the row you want to remove and click **Delete**.

3. To save your work, press CTRL+S or click **Check In**.

Data references from outside the PI System

You can also use PI AF tables to access data that is external to the PI System. Such data might be contained in Microsoft Excel, Access, or SQL Server, or other OLE DB/ODBC data sources. You can either import the table or link to it after you have defined the table structure, as described in Create PI AF tables and Create table column definitions.

Imported tables

PI AF tables with imported data are called imported tables. Imported tables are read/write tables. They are limited in size but are more secure than linked tables. Imported tables are sometimes called internal tables because, unlike linked tables, the table data is managed in PI AF. After the initial import, there is no further relationship between the foreign table and the PI AF table. You can edit the data directly in PI AF.

It is a good practice to limit your imported tables to 10,000 rows of data or less. Imported tables are not designed for storing very large databases. If you need to access a lot of data in PI AF tables, link to external tables instead, which do not present such storage limits. Alternatively, break the table into separate tables when importing.

Linked tables

Linked tables are sometimes called external tables, because the source data is not stored in the PI AF database. You cannot edit an external table from PI AF. Linked tables require additional security configuration because you need to configure how PI AF connects to the external data source. You should set up a reusable table connection, where you configure the type of authentication to access the external table.

Videos

For information on how to import data from external tables, watch this video:

https://www.youtube.com/watch?v=MmXmnac969s&rel=0

For information on how to link to data in external tables, as well as SQL security, watch this video:

https://www.youtube.com/watch?v=JnhodkkyHks&rel=0

Topics in this section

- Authentication for linked tables
- Create reusable table connections
- Access to Microsoft Access or Excel data

- Access to SQL Server data

Authentication for linked tables

When a client application requests external data, the PI AF server queries the external data source and returns the data to the client as a read-only PI AF table.

For externally linked tables, OSIsoft recommends that the OLE DB provider and the PI AF Server have the same bitness (32-bit or 64-bit). To configure an external table connection in PI System Explorer, for example, you would use a PI AF server of the same bitness (typically, 64-bit).

When you configure the linked table, you must specify the credentials that the PI AF server uses to connect to the database. The authentication options are:

- **Impersonate Client**

  If the source database supports Windows authentication, use the Windows identity of the client that is requesting the data. This is an impersonated connection. This is the most secure method of authentication and should be used wherever possible.

- **Supply Password**

  If the source database does not support Windows authentication, or if the database and PI AF server are on different, non-trusted domains, specify a user name and password with the necessary access on the source database. PI AF uses this hard-coded account to read the data in the external data source. For example, MySQL database does not support Windows authentication, so you would use the user name and password of an account on the MySQL database.

  Note:
  You can enter a user ID and password as part of the connection string and save it with a PI AF table connection, regardless of whether support for external PI AF tables for non-impersonated users has been previously enabled (with the `afdiag /DTImp` command).

- **No additional security context**

  This option usually applies when you use Excel or other file-based data sources; otherwise every user needs to be granted read access to the file on the server. With this option, the external table will be accessed using the PI AF Server's identity. In this case, you do not need to specify a username or password when configuring the linked table, nor is Kerberos configuration required.

  Caution:
  Take care to configure SQL security in such a way that the PI AF server's identity does not have more privilege than necessary to retrieve the data. Only PI AF administrators are allowed to configure external tables for security reasons. For that reason, ensure that the PI AF `Administrators` identity and the **Admin** access right are assigned to only a limited set of users when this connection mode is enabled.

Topics in this section

- Restrictions on non-impersonated connections
- Risk of using non-impersonated connections
- Data access recommendations for linked tables

- Security settings for linked tables

Restrictions on non-impersonated connections

Because there are security risks for linked tables that use non-impersonated connections, some PI AF server system administrators restrict or prevent their use. Your system administrator might:

- Require administrative privileges on the PI AF server and write privileges on the PI AF table.
- Prevent creation of linked PI AF tables with non-impersonated connection.
- Prevent creation of any linked tables.

If you have problems with linked tables, consult your system administrator about the PI AF server external table settings.

Risk of using non-impersonated connections

Depending on the configuration of the SQL Server, a user with PI AF administrator privileges could create attacks on the SQL Server and take full control of the system if these following conditions exist:

- A PI AF table is configured to use the PI AF server identity for linking to an external database.
- Non-impersonated linked (external) tables are enabled on the PI AF server.

  By default, non-impersonated linked tables are disabled on the PI AF server. In order for a user to execute an attack, that user would need to enable non-impersonated external tables.
- The PI AF server account has administrative rights on a SQL Server.

  By default, the PI AF server runs under a virtual account, `NT SERVICE\AFService`, and does not have administrative rights to the locally-configured SQL Server or access to remote computer databases. Without administrator rights to the remote database, the possibility for elevation of privilege attacks is limited.

> **Caution:**
> For security reasons, do not grant the PI AF server administrative privileges on the computer or SQL Server when running with non-impersonated queries.

Data access recommendations for linked tables

OSIsoft recommends you observe the following guidelines when you set up linked tables.

- If access to linked tables is not needed, disable it altogether.
- Do not grant the PI AF application service account administrative privileges on the PI AF server or SQL Server when running with non-impersonated queries.
- You must have administrative privileges on the PI AF server to configure an external table that runs non-impersonated queries.

Security settings for linked tables

You use the PI AF Diagnostics utility (**afdiag** located in the `\PIPC\AF` folder) to enable or disable support for external PI AF tables. Since the utility makes a direct connection with the associated SQL Server database, the SQL Server `sysadmin` or `db_AFadmin` role is required.

Use the PI AF Diagnostics utility to adjust security settings for external tables.

Task	Command	Default Setting
Enable support for external PI AF tables	**afdiag /DT**	enabled
Disable support for external PI AF tables	**afdiag /DT-**	
Enable support for external PI AF tables for non-impersonated users	**afdiag /DTImp**	disabled
Disable support for external PI AF tables for non-impersonated users	**afdiag /DTImp-**	
Change security settings for a specific PI AF table	In the Library browser, right-click on the table and click **Security**.	By default, table configuration requires administrative privileges on the PI AF server.
Change security settings for all tables	In the Library browser, right-click on **Tables** and click **Security**.	By default, table configuration requires administrative privileges on the PI AF server.

Note:
You do not have to enable support for external PI AF tables for non-impersonated users. You can include a user ID and password as part of the connection string in a PI AF table connection and check it in, regardless of whether support for external PI AF tables for non-impersonated users is enabled. The defined PI AF table connection can be used within a PI AF table definition. This means that if a SQL Server Login has permission to access the data referenced in the connection string, a PI AF table linked to that PI AF table connection can retrieve the external data.

Create reusable table connections

Create an OLE DB connection and reuse it to configure linked tables from the same data source.

Before you start

Use the 64-bit PI System Explorer to configure connections to linked tables. It includes 64-bit OLE DB drivers, which are required for the 64-bit PI AF Server.

Procedure

1. In the navigator, click **Library**.
2. Choose from the following actions:
   - Select **Table Connections** and click **New Table Connection** on the toolbar.
   - From the browser, right-click **Table Connections** and click **New Table Connection**.

Table connection properties are displayed in the viewer with a default name in the **Name** field.

3. Optional. Edit the default name and add a description for the table connection in the **Description** field.
4. In the **Connection** field, you can either enter a connection string directly, or click **Build** to configure the connection in the Data Link Properties window.
5. In the **Provider** tab of Data Link Properties, choose an OLE DB provider and follow onscreen instructions to configure the connection.

   For help on Microsoft Data Link, click **Help** from any tab, link to the MSDN Resources page, and enter `Data Link Properties Dialog Box` in the search field.
6. In the viewer, select a **Security** option. Some options may not be available.
   - **Impersonate Client** (recommended)
   - Select **Supply Password** and enter a password for the table connection.
   - **No additional security context**

   See Authentication for linked tables for information on these options.
7. On the toolbar, click **Check In** to check in and save the table connection.

Access to Microsoft Access or Excel data

To access data from a Microsoft Access database or a Microsoft Excel workbook, you can link or import the data. The data in a linked table is refreshed when the table is accessed and whenever the time since the last refresh exceeds the table's **Cache Interval** setting.

Imported data is loaded into the PI AF table once. If you ever need to refresh the imported data in a table, you can right-click the table in the Library browser and click **Re-Import Table**.

You can link to or import data from a Microsoft Access table or a Microsoft Excel workbook after you perform the following steps:

- Specify the source database or workbook.
- Create a query that returns the desired data.
- Enter any login credentials required to access the database or workbook.

The exact instructions depend on your hardware configuration and what you are trying to do:

32-bit PI AF server	Follow the instructions in Link or import data.
64-bit PI AF server	• To import, follow the instructions in Link or import data. • To link, follow the instructions in Link to data on a 64-bit PI AF server.

Topics in this section

- Link or import data
- Link to data on a 64-bit PI AF server

Link or import data

These instructions describe how to import data on a 32-bit or 64-bit PI AF server and how to link to data from a 32-bit PI AF server only. To link to data from a 64-bit server, see Link to data on a 64-bit PI AF server.

Procedure

1. In PI System Explorer, navigate to the PI AF table or create one as described in Create PI AF tables.
2. In the **Library** pane, expand the **Tables** node, and click the desired PI AF table.

   The table details display in the right pane.
3. Click **Link** or **Import**.

   The corresponding window opens.
4. Link only: If you are linking the table, enable the **Impersonate Client** option (not displayed for Import).
5. Click **Build**.

   The Data Link Properties window opens.
6. On the **Provider** tab, select the provider according to the version of Microsoft Office that you are using and click **Next.**
   - Office 97-2003: select **Microsoft Jet 4.0 OLE DB Provider**.
   - Office 2007 and higher: select **Microsoft Office 12.0 Access Database Engine OLE DB Provider**.
7. On the **Connection** tab, specify the following and click **OK**.
   - **Data Source**

     The location and file name of the database or workbook (such as `C:\AFTestData.xls`). If you are linking, the path to the file must be relative to the PI AF server.
   - **User Name**

     Login credentials of a user that has been granted read access to the database or workbook.

   > **Note:**
   > To store the password with the connection information, select the **Allow Saving Password** check box. The password is stored as plain text (not encrypted).
8. On the **Advanced** tab, in the **Access permissions** list, select **Share Deny None.**
9. Excel only: On the **All** tab, select the **Extended Properties** value and click **Edit Value**.

   The Edit Property Value window opens.

   Enter the property value according to the version of Microsoft Excel that you are using, and then click **OK**.
   - Excel 97-2003: Excel 8.0
   - Excel 2007 and higher: Excel 12.0

10. To verify that the spreadsheet is accessible, return to the **Connection** tab and click **Test Connection**.

    If the settings are valid, a Test connection succeeded message displays.
11. To dismiss the window and return to PI System Explorer, click **OK**.
12. To define the data to be returned from the spreadsheet, enter an SQL query in the **Query** field. To dismiss the window, click **OK**.
    - Microsoft Excel example: `SELECT * FROM [Sheet1$]`
    - Microsoft Access example: `SELECT * FROM Table1`
13. To review the resulting data, examine the **Table** tab. If the query is specified correctly, the tab contains a table displaying the results.
14. To save your changes, right-click the table node and choose **Check In**.

Link to data on a 64-bit PI AF server

To link to data in an Access database or Excel workbook on a 64-bit PI AF server, you must use the 64-bit Access Database Engine (ACE) data provider; there is no 64-bit Jet data provider.

Procedure

1. In PI System Explorer, navigate to the PI AF table or create one as described in Create PI AF tables.
2. In the **Library** pane, expand the **Tables** node, and click the desired PI AF table.

   The table details display in the right pane.
3. Click **Link**.

   The corresponding window opens.
4. Enable the **Impersonate Client** option.
5. In the **Connection** field, enter a valid connection string for the Excel workbook or Access database, using the Microsoft Office 12.0 Access Database Engine OLE DB Provider (must be installed on the PI AF server), as in the following examples:
   - Microsoft Excel example:
     ```
     Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\example.xlsx;Extended
     Properties="Excel 12.0 Xml";
     ```
   - Microsoft Access example:
     ```
     Provider=Microsoft.ACE.OLEDB.12.0;Data Source=c:\example.accdb;Persist
     Security Info=False;
     ```
6. To define the data to be returned from the spreadsheet, enter an SQL query in the **Query** field. To dismiss the window, click **OK**.
   - Microsoft Excel example: `SELECT * FROM [Sheet1$]`
   - Microsoft Access example: `SELECT * FROM Table1`
7. To review the resulting data, examine the **Table** tab. If the query is specified correctly, the tab contains a table displaying the results.
8. To save your changes, right-click the table node and choose **Check In**.

Access to SQL Server data

When you import data from or link to data in a Microsoft SQL Server table, you must define valid connection information. The steps for linking or importing depend on the connection method that you choose, as described in Authentication for linked tables.

Topics in this section

- Use Windows impersonated security connection
- Use Windows non-impersonated security
- Use SQL Server security
- Configure security on the target table's database
- Import or link SQL server tables

Use Windows impersonated security connection

Link to or import data from a SQL Server table with the **Impersonate Client** option. See Data access recommendations for linked tables.

> **Note:**
> If linking to a SQL Server that is not on the same computer as the PI AF server, it may be necessary to configure Kerberos to allow the client's identity to be forwarded from the PI AF server to the SQL Server.

Procedure

1. Create a local user group for impersonated clients.
2. Configure security on the target table's database.
3. Import or link SQL server tables.

Create a local user group for impersonated clients

If the table to which you want to connect resides in a SQL Server instance other than the one where the PI AF SQL database (PIFD) resides, ensure that the table is accessible via Windows authentication.

Procedure

1. On the computer where the SQL Server instance resides, click **Start** > **Administrative Tools** > **Computer Management**.

   The Computer Management application starts.
2. Expand **Local Users and Groups**.
3. Right-click **Groups** and choose **New Group**.
4. In the New Group window, create a local user group for the users that require access to the database table.
5. Add to this group the accounts of all users that might be impersonated.
6. Click **OK** to add the selected users, and then click **Close** to dismiss the New Group window.
7. Close the Computer Management application.

Use Windows non-impersonated security

Link to or import from a SQL Server table with the `no additional security` context option.

Procedure

1. Create a local user group for PI AF application service account
2. Configure security on the target table's database
3. Import or link SQL server tables

Create a local user group for PI AF application service account

If the table to which you want to connect resides in a SQL Server instance other than the one where the PI AF database (PIFD) resides, ensure that the table is accessible to the PI AF application service account.

Procedure

1. On the computer where the SQL Server Instance resides, click **Start** > **Administrative Tools** > **Computer Management**.

   The Computer Management application starts.
2. Expand **Local Users and Groups**.
3. Right-click **Groups** and choose **New Group**.
4. In the New Group window, create a local user group to hold the identity of the PI AF Server Application Service.
5. Add the account of the user associated with the PI AF Server Application Service to the new group. If the PI AF Server Application Service is running under the `NT AUTHORITY\NetworkService` account, then add the PI AF server's computer account to this group.

   Note:
   If the PI AF Server Application Service is running as the Local System or Local Service account, you most likely need to use SQL Server authentication (SQL Server and Windows authentication mode) instead of Integrated security.
6. Click **OK** to add the selected user.
7. Close the Computer Management application.

Use SQL Server security

Link or import data from a SQL Server table using SQL Server authentication.

Before you start

If you are connecting to a remote SQL Server instance, ensure SQL Server is configured to accept Remote Connections.

If you are using a SQL Server account, ensure the SQL Server instance is configured to allow mixed mode authentication.

Procedure

1. Create a SQL Server user.
2. Configure security on the target table's database.
3. Import or link SQL server tables.

Create a SQL Server user

If the target table (the table to which you want to connect) resides in a SQL Server instance other than the one in which the PI AF database (PIFD) resides, create a user and enable database access for the user as described in this topic.

Procedure

1. Open Microsoft SQL Server Management Studio and connect to the SQL Server Instance that contains the target table.
2. Under the SQL Server Instance, expand the Security folder and then expand the Logins folder.
3. Create a new Login and enter a name in the **Login Name** field.
4. Select the **SQL Server Authentication** option.
5. Enter the password in the **Password** and **Confirm Password** fields.
6. From the **Default Database** list, select the database that contains the target table.
7. Select the **User Mapping** page.
8. Select the row for the Database that contains the target table.
9. Select the **Map** check box for the selected database.
10. Click **OK** to close the Login – New window and save the new Login.
11. Expand the **Databases** folder, then the folder for the target database, and grant the necessary permission to execute the query that will be used to the Login just created.

    For example, if the query that will be used is a SELECT statement that specifies a single table, expand the **Tables** folder for the target database, expand the **Tables** folder, then right-click the table to which the query refers and choose **Properties**.
12. In the Table Properties window, select the **Permissions** page, then the Login, then **Grant the Login the Select Permission**. Click **OK** to close the Table Properties window.
13. Close Microsoft SQL Server Management Studio.

Configure security on the target table's database

Any account under which the specified query might be executed needs to be granted permission to execute that query.

Procedure

1. Open Microsoft SQL Server Management Studio and connect to the SQL Server Instance that contains the target table.
2. Under the SQL Server Instance, expand the **Security** folder, and then expand the **Logins** folder.

3. Right-click the **Logins** folder and choose **New Login**.
4. Use the **Search** button to find the group created in the previous section and choose that group for the Login name.
5. Select the **Windows authentication** option, and select the database that contains the target table as the `Default database`.
6. Select the User Mapping page.
7. Select the row for the Database that contains the target table.
8. Select the **Map** check box for the selected database.
9. Expand the **Databases** folder, then the folder for the target database, and grant the necessary permission to execute the query that will be used to the Login just created.

   For example, if the query that will be used is a `SELECT` statement that specifies a single table, expand the **Tables** folder for the target database, expand the **Tables** folder, then right-click the table to which the query refers and choose **Properties**.
10. In the Table Properties window, select the **Permissions** page, search for and select **Login**, then **Grant the Login the Select Permission** and click **OK** to close the Table Properties window.
11. Close Microsoft SQL Server Management Studio.

Import or link SQL server tables

Whether you are importing from or linking to Microsoft SQL Server tables, the process is essentially the same. The following instructions describe how to link to an existing PI AF table using the PI System Explorer.

Procedure

1. To browse to the target PI AF table, display the **Library** pane, expand the **Tables** node, and click the desired table.

   Table properties display in the right pane.
2. Click **Link**.

   The Table Link window opens.
3. Click **Build**.

   The Data Link Properties window opens.
4. On the **Provider** tab, select **SQL Server Native Client**.
5. On the **Connection** tab, configure the SQL Server instance that contains the database to which you want to connect.
6. Configure authentication:
   - For Integrated security, select the `Use Windows NT Integrated Security` option.
   - For SQL Server security, select the `Use a Specific User Name and Password` option. Then, enter the SQL Server Login name in the **User Name** field and enter the password in the **Password** field.

> **Note:**
> Ensure that support for external PI AF tables for non-impersonated users has been enabled with the **`afdiag /DTImp`** command.

7. Select the `Select the Database` option and choose the database that contains the table to which you want to connect.
8. To verify that connection settings work, click **Test Connection**. If the settings are correct, PI System Explorer displays a success message.

> **Note:**
> Test Connection verifies that the account with which the PI System Explorer is running has access to the specified database. However, if you choose Integrated Security in the Table definition and choose the `no additional security context` option for table security, the account associated with the PI AF Server Application Service is used to connect when a user displays the data by viewing the **Table** tab.

9. Click **OK**.
10. In the **Query** field, specify the SQL query that returns the desired data and click **OK**.

    If the connection string requires a User ID and Password, click the supply password button and enter the SQL login password used to test the connection. This replaces the value in the **Password** field.
11. To display the data retrieved by the query, view the **Table** tab.

Configure table lookup data references

Create a table lookup data reference when you want an attribute to obtain its value from a table column.

Procedure

1. In the Elements or Library browser, select the desired element or element template.
2. In the viewer, select the attribute or attribute template for which you want a table lookup value.
3. In the **Data Reference** field in the palette, choose **Table Lookup** from the list.
4. Click **Settings**.
5. In the Table Lookup Data Reference window, select a previously defined table from the **Table** drop-down list. You can also choose from the following options:
   - Click (Manage Tables) to open a list of tables you can search or filter. To select a table, highlight it in the list and click **OK.**
   - Click (Table Properties) to view or edit properties for the selected table.
   - Click (Create New Table) to define a new table.
6. From the **Result column** list, select the column in the table from which you want to read the value.

> **Note:**
> Select the **Stepped** check box for the value to be stepped when plotted in a trend. With this setting, there is no interpolation between the table values.

7. From the **Unit of Measure** list, select the appropriate unit of measure in which the data in the result column is stored.

   Note:
   It is preferable to set the UOM directly in the table definition so that you do not need to specify it with each table lookup.

8. In **Time Zone**, you can define a setting if it has not previously been set in the table or column definition.

9. From the **Rule** list, choose an option:

   - `Select first row matching criteria`

     Use the **Order by** list to specify the sorting order. This order is used to select a row when more than one row matches the criteria. For more information, see Select first row matching criteria.

   - `Summarize all rows matching criteria`

     Select a summary operation from the **Summary** list to perform the selected operation on the selected column over the range of rows that match the criteria. For more information, see Summarize all rows matching criteria.

   - `Table provided time series data`

     Choose this option if the table has values with associated time stamps and you wish to treat these values as time series data. From the **Time Column** list, select the table column that contains the time stamps you want to use. Only columns with a value type of DateTime are listed. The `WHERE` clause is not required when you choose this option. For more information, see Table provided time series data.

     Note:
     To ensure that columns with DateTime value types are displayed, only specify time zone settings on the **General** tab of the overall table definition. Do not specify time zones on the column definition (on the **Define Table** tab).

10. In the **Where** pane, use the menus and buttons to build the table query.

    Note:
    You can manually type the entire clause into the **Complete WHERE Clause** text field. See WHERE clause syntax for more information.

    - From the **Column** list, select the column of the table to use in the query.
    - From the **Operator** list, select the relational operator to use in the query.
    - From the **Attribute or Value** list, select an attribute or a literal value to use in the query.
    - Click **Add And** or **Add Or** to write the `WHERE` clause into the **Complete WHERE Clause** field with an `AND` or an `OR` operator.
    - Edit the clause in the **Complete WHERE Clause** field as needed.

    Note:
    The **Add And** or **Add Or** buttons automatically generate the necessary syntax, UOM, and time zone conversions when possible.

11. Optional. Edit values for table parameters.

Table parameters apply only to linked tables. For more information, see Parameters for linked table queries.

12. Optional. For **Replacement Values**, choose attributes or literal values to return when the table query cannot find a matching row or encounters a null result.

Topics in this section

- Select first row matching criteria
- Summarize all rows matching criteria
- Table provided time series data
- WHERE clause syntax
- Parameters for linked table queries

Select first row matching criteria

The `Select first row matching criteria` rule enables you to specify the first row that matches the value returned from the `WHERE` clause.

Syntax

```
SELECT column FROM table WHERE where clause ORDER BY column ASC|DESC; options
and parameters
```

Arguments

- **`SELECT` *`column`***

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  In addition to `=`, `<>`, `>`, `>=`, `<`, `<=`, `LIKE`, and `IN` relational operators, you can specify `INTERPOLATE` to interpolate a value for the result column based on an interpolation of the specified input columns.

  Beginning with PI AF 2018, you can perform two kinds of interpolation:

  - Linear interpolation, in which interpolation is for 2 columns of data. To specify linear interpolation, use `INTERPOLATE(`*`column`*`,` *`value`*`)` syntax.
  - Standard bilinear interpolation, in which interpolation is for 3 columns of data. To specify bilinear interpolation, use `INTERPOLATE(`*`column_X`*`,` *`value_X`*`) AND INTERPOLATE(`*`column_Y`*`,` *`value_Y`*`)` syntax.

  Irregular bilinear interpolation and bilinear extrapolation are not currently supported.

  Note:
  The `INTERPOLATE` operator also returns values from outside the table range.

For more information on `WHERE` clause syntax, see WHERE clause syntax. For examples of `INTERPOLATE` operator usage, see Examples of linear and bilinear interpolation in table lookup data references.

- **`ORDER BY` *`column`***

  Optional. Specifies the sorting order so that the correct row is used when more than one row matches the `WHERE` clause. Ascending (**`ASC`**) order is the default unless descending (**`DESC`**) is specified.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Specifies the time zone of the source table. **Note:** OSIsoft recommends you set the time zone in the general description of the table, so that you do not need to specify it with each table lookup.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of `No Data` is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of `No Data` is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

Example

```
SELECT [Installation Date] FROM [Equipment Specifications] WHERE [Asset ID] =
'%Element%'
```

Examples of linear and bilinear interpolation in table lookup data references

Linear interpolation example

In a tank strapping table that contains a `Level` column and a `Volume` column, the following configuration string interpolates the volume based on the level reading:

```
SELECT Volume FROM MyTable WHERE INTERPOLATE(Level, @MyLevelReading)
```

Assume the sample table has the following rows:

Level	Volume
1	0.0
2	20.0
3	30.0
4	40.0
5	60.0
6	70.0

A `Level` reading of 2.2 results in a returned `Volume` of 22.0.

Bilinear interpolation example

In an air velocity table that contains an `X Horizontal Position` column, a `Y Vertical Position` column, and a `Velocity` column, the following configuration string interpolates the velocity based on the X horizontal and Y vertical positions.

```
SELECT Velocity FROM Table1 WHERE INTERPOLATE([X Horizontal Position]), @X) AND
INTERPOLATE([Y Vertical Position]), @Y)
```

Assume the sample table has the following rows:

X Horizontal Position	Y Vertical Position	Velocity
180	140	4.6
220	140	4.1
260	140	2.7
180	180	4.8
220	180	4.4
260	180	2.5
180	220	4.5
220	220	4.4
260	220	2.5

An `X Horizontal Position` of 245 and a `Y Vertical Position` of 165 results in a returned `Velocity` of 3.2171875.

Summarize all rows matching criteria

The `Summarize all rows matching criteria` rule enables you to perform a summary operation on the rows in a table column that match your selection criteria.

Syntax

```
SELECT summary(column) FROM table WHERE where clause; options and parameters
```

Arguments

- **`SELECT summary(column)`**

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets. You can select one of the following summary operations.

Operation	Description
Sum	The total of all row values.
Avg	The average of all row values.
Min	The minimum value of all rows.
Max	The maximum value of all rows.
Count	The number of rows.
StDev	The extent of deviation for all row values.
Var	The average measure of how far all row values differ from the mean.
None	When no operation is specified: ◦ If the result attribute is not an array, the value of the selected column in the first row that matches the `WHERE` clause is returned. ◦ If the result attribute is an array, an array with one value from each column of all rows that match the `WHERE` clause is returned.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  For more information on `WHERE` clause syntax, see WHERE clause syntax.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Specifies the time zone of the source table. **Note:** OSIsoft recommends you set the time zone in the general description of the table, so that you do not need to specify it with each table lookup.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of No Data is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of No Data is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

Table provided time series data

The `Table provided time series data` rule enables you to select a table containing values with a DateTime data type, and specify the column that contains time series data.

Syntax

```
SELECT column FROM table WHERE where clause; TC=column; options and parameters
```

Arguments

- **`SELECT` *`column`***

  If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`FROM` *`table`***

  If a table name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`WHERE` *`clause`***

  Optional. For more information on `WHERE` clause syntax, see WHERE clause syntax.

- **`TC` *`column`***

  Specifies the column that contains time series values. If a column name contains non-alphanumeric characters, including spaces, it must be enclosed in [ ] brackets.

- **`Options`**

  You can enter the following options in a list separated by semicolons.

Stepped	When set to True, the returned value plots as stepped in applications.
TZ=*time zone*	Not supported at the column level. You need to specify the time zone in the **Time Zone** field on the **General** tab for the overall table.
UOM=*uom*	Specifies the unit of measure for the value returned by the result column. **Note:** You can also set the unit of measure in the table column definition, so that you do not need to specify it with each table lookup.
RWM=*value*	Specifies the value to return when there is no column match. If the value is `No Data`, the digital state of No Data is returned.
RWN=*value*	Specifies the value to return when the result column is null. If the value is `No Data`, the digital state of No Data is returned.

- **`Parameters`**

  You can enter parameters in a list separated by semicolons. Begin each parameter name with the @ character in @*parameter*=*value* format (*value* is described in "Attribute or Value" in WHERE clause syntax). For additional information on using parameters, see Parameters for linked table queries.

WHERE clause syntax

The `WHERE` clause uses SQL syntax and conforms in general to the syntax described on the MSDN DataColumn.Expression Property (https://msdn.microsoft.com/en-us/library/system.data.datacolumn.expression(v=vs.110).aspx) page. You can type the clause directly into the **Complete WHERE Clause** field in the Table Lookup Data Reference window, or allow PSE to provide the correct syntax by using the **Column**, **Operator**, and **Attribute or Value** lists in the **Where** section. The `WHERE` clause is optional for the `Table provided time series data` rule.

The `WHERE` clause syntax follows these guidelines.

Column	Column names that contain non-alphanumeric symbols must be surrounded by brackets: `[Asset ID] ='%Element%'`
Operator	• The `INTERPOLATE` operator is available with the `Select first row matching criteria` rule. • The comparison operators `=`, `<>`, `>`, `>=`, `<`, `<=`, `IN`, and `LIKE` are supported for all lookup rules. You can use the * and % characters interchangeably for wildcard characters in a `LIKE` comparison. A wildcard is allowed at the start and end of a pattern, or at the end of a pattern, or at the start of a pattern, but not in the middle of a string. If the string in a `LIKE` clause contains a * or %, those characters should be enclosed in [ ] brackets.
Attribute or Value	*@attribute* Returns the value of the PI AF attribute. The attribute must be enclosed in [ ] brackets if it contains any non-alphanumeric character, including spaces, or includes a UOM or Time Zone specification. For example: `[height] >= @[Level Gauge;UOM=m]`
	literal • Strings should be enclosed in single quotes. • Numeric values are not quoted and should be in invariant format (where the decimal character is '.'). • Timestamps are best specified in `yyyy.mm.dd hh.mm.ss` format (0-23 for `hh`), and enclosed in the # character. For example: `#2015.01.30 14:00:00#`
	substitution Select or enter a substitution parameter in `'%substitution parameter%'` format. For more information on substitution parameters, see List of PI AF substitution parameters.

Parameters for linked table queries

The query for a linked table determines which data from an external source to include in the table. You can include parameters in the query, and, as you configure a table lookup data reference that uses the linked table, you can specify parameter values. This enables you to use a single linked table to obtain different results in every table lookup data reference that uses it.

Using parameters in a linked table query is useful, for example, to limit the number of rows returned from a very large external table. You can add conditions and parameters to return more targeted results, such as all rows that include a device or manufacturer ID number, specific for each table lookup data reference.

As you configure a linked table in the Table Link window, you can add table parameters to its query and set default values for them. Then, in the Table Lookup Data Reference window, as you define a data reference using the linked table, you can enter table parameter values specific for that data reference. You can also supply parameter values programmatically using AF SDK. Parameter values can be specific values or come from other attribute values or from pre-defined substitution variables, such as *%Element%*.

Add table parameters to a linked table query

As you define a linked table in the Table Link window, you can add parameters to the table query and set default values for them.

Procedure

1. Edit the text in the **Query** box to include the new parameter(s). Parameter names must begin with the @ character.
2. Click inside the **Parameters** table to display the new parameters from the query.
3. Enter default values for each parameter to determine the default results from the query and click **OK**. After you have added parameters to the query, you can specify values for them as you configure a table lookup data reference that uses the linked table.

Example

Consider the following query for a linked table named `MyTable`. The WHERE clause limits the selection from an external table (`BigTable`) to those rows with a particular `RowID`:

```
SELECT * FROM BigTable WHERE RowID = 101
```

Replace the fixed value 101 with a table query parameter `@id` (note that query parameter names must begin with the @ character):

```
SELECT * FROM BigTable WHERE RowID = @id
```

Now, for every table lookup data reference that uses `MyTable`, you can supply different table parameter values for `@id` to get different results from the query.

For example, in the Table Lookup Data Reference window, as you configure a data reference, enter `@AssetID` for the value of `@id` in the **Table Parameters** list. This sets `@id` to the current value of the attribute `AssetID`. The corresponding query for this would be:

```
SELECT Result FROM MyTable; @id=@AssetId
```

This query returns rows whose `RowID` matches the current value of `AssetID`.

URI Builder data references

In previous versions of PI Notifications, you could define a web link or a link to a client application such as PI WebParts or PI Vision as content within a notification. However, because the links were disconnected from the elements and attributes in PI System Explorer, only users

with PI Notifications installed were able to go find objects referenced in the email notification itself. Also, you could only include dynamic content as a value of a parameter.

With URI Builder in PI AF 2016, you can configure data references that create dynamic links on event frames, transfers, elements or models, and include the same attribute values and substitution parameters as other data references such as String Builder or Formula. URI Builder conforms to standard syntax rules (described in the Requests for Comments document RFC 3986 URI Generic Syntax (https://tools.ietf.org/html/rfc3986)) and automatically handles the escaping of characters which are complex and cumbersome to enter manually. It supports generic web links (that use http and https schemes), as well as PI Vision links.

You can also use PI Notifications 2016 to send the link to other users via electronic mail, an instant messaging client, or a client application such as PI Vision.

Create URI Builder data references

Copy an existing web link or PI Vision link and make it dynamic in a URI Builder data reference in which you substitute part of the URI with a value derived from an attribute, an attribute reference or a substitution parameter. You assign the dynamic web link value or PI Vision link value to an attribute template or attribute.

Before you start

When a URI Builder data reference is used in a template, value substitution takes place at run time.

Procedure

1. Open a web browser and locate the web link or the PI Vision link you use as a starting point in creating a link value on an attribute template or attribute.
2. Select the entire URI and copy it.
3. In the navigator pane of PI System Explorer, choose from the following actions.

To create a URI data reference on ...	Do this ...
A template	a. Click **Library**. b. Expand **Templates** in the browser tree. c. Choose a template type and select an existing template, or click **New Template**.
An element	a. Click **Elements**. b. Select an existing element in the browser tree, or click **New Element**.
A model	a. Click **Elements**. b. Select an existing model in the browser tree, or right-click and click **New Model**.
An event frame	a. Click **Event Frames**. b. Expand a search collection for event frames in the browser tree. c. Select an existing event frame, or click **New Event Frame**.

A transfer	a. Click **Event Frames**. b. Expand a search collection for transfers in the browser tree. c. Select an existing transfer or click **New Transfer**.

4. Choose from the following actions.
   - In the **Attribute Templates** tab, select an existing attribute template, or click **New Attribute Template** to create a new attribute template.
   - In the **Attributes** tab, select an existing attribute, or click **New Attribute** to create a new attribute.
5. In the attribute configuration panel, configure the attribute template or attribute as needed, and choose **URI Builder** from the **Data Reference** drop-down list.
6. Click **Settings**.
7. In the URI Builder Data Reference window, click in the **Paste a URI from a web browser to use as a template for configuration** field and paste the URI you copied in step 2.
8. Click **Continue**.

   The pasted URI is displayed in the URI Builder Data Reference window, broken down into its **Scheme**, **Address or Host**, **Port**, and **Path** field components. If the URI contains queries and fragments, those are also displayed in the **Query** and **Fragment** fields.
9. In the URI Builder Data Reference window, choose one of the following actions.

To configure ...	Do this ...
A dynamic web link	Modify the components of the pasted URI as needed. ◦ You can choose between **Scheme** values of http and https. ◦ You can change the **Port** value to any number between 1 and 65535. The default values for http and https schemes are 80 and 443 respectively. ◦ To make portions of the **Path** field dynamic, select a segment. Then click and select from the following options: ▪ A list of other attributes for the selected object. Note that attributes must appear in single quotes. ▪ A list of attribute references. You can also select **Search** to enter search criteria in the Attribute Search window. ▪ A list of commonly used substitution parameters. For more information, see List of PI AF substitution parameters. ◦ In the **Query** table, you can modify **Key** and **Value** settings as needed. ▪ To add a key, click . In the **Key** and **Value** fields, type a value or click and select from the same options as above. ▪ To delete a parameter, select a key or value and click .

A PI Vision link	a. Click the **PI Vision** option. b. In the **Display** field, modify the value as needed. You can also click ▸ and select from the following options: ▪ A list of other attributes for the selected object. Note that attributes must appear in single quotes. ▪ A list of attribute references. You can also select **Search** to enter search criteria in the Attribute Search window. ▪ A list of commonly used substitution parameters. For more information, see List of PI AF substitution parameters. c. In the **Parameters** table, modify **Key** and **Value** settings as needed. ▪ To add a key, click ✨. In the **Key** field, click ▸, select a parameter and assign a value. In addition to typing a value you can select from the same options as in step b above. ▪ To delete a parameter, select a key or value and click ✕. When linking to a display on an event frame, URI Builder automatically adds Asset, Start Time and End Time parameters. For more information on other parameters, see the PI Vision topic "URL parameters reference" in Live Library (https://livelibrary.osisoft.com).

10. Click **OK**.

Units of measure in PI AF

PI AF ships preloaded with numerous standard unit-of-measure classes and conversion factors. You can extend these classes by adding new units of measure, as well as new measurement classes. The implementation of the units of measure (UOM) feature in PI AF is based on the International System of Units (SI).

UOM database

All PI AF databases use the same set of UOMs, which are defined in the UOM database. All UOMs have the Read permission set, but other permissions can be set for the UOM database as a whole, as described in Configure security for the UOM database.

Beginning with PI AF 2017 R2, users with administration privileges can review audit trail data on the UOM database by selecting **File** > **Audit Trail Events** after they have clicked **Unit of Measure** in the navigator. For more information, see Track PI AF changes with Audit Trail.

UOM classes and canonical units

In PI AF, each unit of measure is based on a UOM *class*. Classes represent measurable properties, such as temperature, length, time, and mass. Each class has a *canonical unit*. This is the base unit from which PI AF converts values to other units when required. For example, the canonical unit for the `Length` class is `meter`, and the abbreviation is `m`.

UOM conversions in client applications

The UOM feature enables automatic unit conversions in client applications. For example, suppose a PI AF attribute has a UOM of `meter`. A PI ProcessBook user who is viewing that attribute value can choose to view the value in a different unit, such as `foot`. PI AF automatically converts the data from meters to feet.

UOM abbreviations

Prior to PI AF 2015 R2, all UOM abbreviations were case insensitive. A new option was added in PI AF 2015 R2 (v2.7.5) that allowed users to configure the PI AF server to support case-sensitive UOM abbreviations. Case-sensitive UOM abbreviations enable you to define abbreviations that differ only by case, such as `MV` (megavolt) and `mV` (millivolt), or `s` (second) and `S` (Siemens).

Note:
Beginning with PI AF 2017 R2, UOM case-sensitivity is enabled by default in a new install, but during an upgrade the current case-sensitivity setting remains unchanged. When case-sensitivity is enabled, only PI AF 2015 R2 or later clients can connect to that server.

Time integral UOMs

Beginning with PI AF 2017, you can view the UOMs associated with a time-weighted total (**Automatic** and **Per-Day**) of rate UOMs. Rate UOMs are from classes that contain time in the denominator of their **Base Units of Measure** (for example, Mass Flow Rate contains $\texttt{Time}^{-1}$, Pressure contains $\texttt{Time}^{-2}$). Each time-weighted total property displays the name of the UOM that will result from a summary Total (**Per-Day**) or TotalWithUOM (**Automatic**) data call.

UOM groups

Beginning with PI AF 2017 R2, UOM groups such as **Metric**, and **US Customary** are available so that you can map UOMs in a UOM class to the setting that is most appropriate for your location. By selecting the **Display UOM Group** option in **Tools** > **Options**, you can then see attribute values displayed in the preferred unit of measure instead of the value selected in an attribute template's **Default UOM** field. For more information on the US Customary UOM group, see the Wikipedia article United States customary units (https://en.wikipedia.org/wiki/United_States_customary_units).

Video

For information on how to edit units of measure, watch this video:

https://www.youtube.com/watch?v=YTG3Dk_SxNM&rel=0

Topics in this section

- Case sensitivity of UOM abbreviations
- Base classes and derived classes
- Create UOM classes
- Create units of measure
- Calculate conversion values for a UOM
- UOM groups
- Origin of UOMs

Case sensitivity of UOM abbreviations

A PI AF Client that is older than PI AF 2015 R2 (v2.7.5) cannot connect to a PI System that has been enabled to use case-sensitive UOM abbreviations. This is due to the possibility of UOM abbreviations being misinterpreted: they are often stored in data references, analysis rules, and client application files, such as spreadsheets and PI ProcessBook displays.

Note:
You must upgrade a PI AF Client to PI AF 2015 R2 or later before it can use case-sensitive UOMs.

References to UOM abbreviations

When UOM case-sensitivity is enabled in PI AF 2015 R2 or later, any stored reference to a UOM abbreviation requires that the correct case for the abbreviation be used. For example:

- Data references that are configured with UOMs (such as Formula data references) must use the correct abbreviation, or else the UOM lookup fails.
- Analysis rules, such as the Convert function, fail if configured with incorrect casing: `Convert('x', 'Kg')` instead of `Convert('x', 'kg')`.
- Applications or data references that programmatically access the UOM database must use the correct casing.

Note:
You cannot use the case-sensitivity option in the PI System to alert you to any configured uses that contain incorrect casing.

UOM names

Unlike abbreviations, UOM names remain case-insensitive in PI AF 2015 R2 or later when UOM case-sensitivity is enabled. However, when you create a UOM abbreviation, it cannot match the name of a different UOM, even if it differs in casing.

For example, the following is valid:

Name	Abbreviation
UOM1_name	UOM1
UOM2	uom1
UOM3	Uom1

The following is invalid and generates the error message: `'uom1' already exists in Unit of Measure 'UOM1' in Unit-of-Measure Database.`

Name	Abbreviation
UOM1	uom
UOM2	uom1

Configuration of case-sensitive UOM abbreviations

Although case-sensitive UOM abbreviations are enabled by default in new PI AF 2017 R2 installations, for older PI AF server versions you can change the setting for case-sensitive UOMs from the disabled state with the PI AF Diagnostics command-line utility (`afdiag`).

For more information on the `afdiag` utility, see PI AF Diagnostics utility.

Enable case-sensitive UOM abbreviations

To enable case-sensitive UOM abbreviations, you enter `afdiag /ucs` on the command line.

A new UOM, `millivolt` with a UOM abbreviation of `mV`, is added to the UOM database provided that there is no conflict with existing units of measure.

Disable case-sensitive UOM abbreviations

You can reset the PI AF server to case-insensitive UOM abbreviations with the same `afdiag` utility.

To disable case-sensitive UOM abbreviations, you enter `afdiag /ucs-` on the command line.

If any existing UOM abbreviations differ only by case, the utility does not allow the feature to be disabled and displays a list of the conflicting abbreviations. At a minimum, you need to remove the millivolt unit of measure that was created when you enabled case-sensitive UOM abbreviations, because `millivolt` (`mV`) conflicts with `megavolt` (`MV`).

Base classes and derived classes

PI AF has predefined a small set of *base classes*, and a larger set of classes that are derived from the base classes, called derived classes. Derived classes are simply classes that can be expressed in terms of the UOM base classes. For example, both the `Area` class and the `Volume` class are derived from the `Length` class. You can define virtually all units of measure in terms of a small group of UOM base classes.

PI AF uses the canonical unit for the base class to determine the canonical units for all classes derived from that base class. You cannot change the canonical unit for a base class or a derived class. For example, since the `Area` class is based on the `Length` class, the canonical unit for `Area` is the square meter (`m²`).

PI AF includes numerous standard UOMs, UOM classes, and conversion factors, but you can always add new ones. It is not typically necessary to create a new base class, although you might occasionally want to add one for non-physical measurements. For example, you could add a class called `Bytes` that could include UOMs of `bytes`, `KB`, `MB`, `GB`, and so on.

Topics in this section

- UOM base classes
- Common UOM derived classes

UOM base classes

The following predefined UOM base classes are available in PI AF.

Class	Canonical unit
Electric Current	ampere (A)
Length	meter (m)
Mass	kilogram (kg)
Moles (amount of substance)	mole (mol)
Plane Angle	radian (rad)
Quantity	count
Ratio	percent (%)
Temperature	kelvin (K)
Time	second (s)

Common UOM derived classes

The following predefined UOM derived classes are available in PI AF.

Class	Based on ...	Canonical unit
Angular Velocity	Plane Angle * $Time^{-1}$	radian per second (rad/s)
Area	$Length^{2}$	square meter (m^2)
Density	$Mass*Length^{-3}$	kilogram per cubic meter (kg/m^3)

Class	Based on ...	Canonical unit
Electric Charge	Electric Current * Time	coulomb (C)
Electric Potential	Electric Current^{-1} * Length^{2} * Mass * Time^{-3}	volt (V)
Energy	Length^{2} * Mass * Time^{-2}	joule (J)
Force	Mass * Length * Time^{-2}	newton (N)
Frequency	Time^{-1}	hertz (Hz)
Mass Flow Rate	Mass * Time^{-1}	kilogram per second (kg/s)
Power	Mass * Length^{2} * Time^{-3}	watt (W)
Pressure	Mass * Length^{-1} * Time^{-2}	pascal (Pa)
Specific Energy, Specific Enthalpy	Length^{2} * Time^{-2}	joule per kilogram (J/kg)
Speed	Length * Time^{-1}	meter per second (m/s)
Volume	Length^{3}	cubic meter (m^{3})
Volume Flow Rate	Length^{3} * Time^{-1}	cubic meter per second (m^{3}/s)

Create UOM classes

Create UOM classes when you need additions to the predefined classes in the UOM database.

Procedure

1. In the navigator, click **Unit of Measure**.
2. Choose from the following actions.
   - On the toolbar, click the **New Class** icon.
   - Right-click in the Unit of Measure browser and select **New Unit-of-Measure Class**.
3. In the Unit-of-Measure Class Properties window, fill out the properties on the **General** tab.
   a. In the **Name** field, enter a name for the class. See Valid characters in PI AF object names, if necessary.
   b. Optional. In the **Description** field, provide a description of the UOM class.
   c. In the **Canonical UOM** field, type the name of the canonical UOM for this class. If the UOM does not exist, PI AF creates it when it creates the new class.
   d. In the **Canonical UOM Abbreviation** field, enter a unique abbreviation.
   e. In the **Base Units of Measure** field, enter a calculation for the unit of measure. Choose from the following actions.

To ...	Do this ...
Add a base class calculation	a. Click . b. In the Add Base UOM Class window, select a base class from the **Base UOM Classes** list. c. In the **Base Power** field, type the exponent value or click one of the arrow keys until the exponent you want is displayed. d. Click **OK**. e. Repeat these steps as needed, to create a multiplication (positive base power) or division (negative base power) calculation.
Remove a base class calculation	a. Select a base class calculation entry. b. Click . c. In the Delete confirmation window, click **Yes**.

Note:
If you are defining a base class, leave the **Base Units of Measure** field blank.

4. Optional. You can add new units of measure to the class you are defining.
   a. Click the **Units of Measure** tab.
   b. Right-click in the list and select **New Unit of Measure**.
   c. In the Unit of Measure Properties window, define the new unit of measure. For more information, see Create units of measure.
5. Click **OK**.

Create units of measure

You create units of measure for a UOM class in the UOM database.

Procedure

1. In the navigator, click **Unit of Measure**.
2. In the Unit of Measure browser, select the class to which you want to add a new unit of measure.
3. Choose from the following actions.
   - On the toolbar, click **New UOM**.
   - Right-click the class in the browser and select **New Unit of Measure**.
   - Right-click in the UOM viewer and select **New Unit of Measure**.
4. In the Unit of Measure Properties window, enter a unique name for the unit of measure in the **Name** field.

   Note:
   The name cannot be the same as a previously defined UOM abbreviation or UOM name.
5. In the **Abbreviation** field, enter a unique abbreviation for the unit of measure.

If UOM case-sensitivity has been enabled, you can use the same abbreviation as a previously defined UOM abbreviation, so long as the case is different. For example, you can use `FR`, `fr`, `Fr`, and `fR` for four different UOM abbreviations.

> **Note:**
> The abbreviation cannot be the same as a previously defined UOM name, regardless of case.

6. Optional. In the **Description** field, enter a description for the unit of measure.
7. In the **Reference UOM** field, select the unit of reference class from which the new unit can be converted. The default value is the same as the read-only **Canonical UOM** value.
8. Choose one of the following conversion methods.

> **Caution:**
> Conversions that are based on **Formula** have some limitations as well as impact on performance. You should only select **Formula** when **Simple** conversions are inadequate.

Option	Description
Simple	Converts a UOM based on scaling factor and base offset values that you enter in the following fields. ◦ In the **Factor** field, enter the conversion factor from the reference UOM to the new UOM. For example, °C has a factor of 1 relative to kelvin. ◦ In the **Offset** field, enter a conversion offset from the reference UOM value. For example, °C has an offset of 273.15 from kelvin. **Note:** Although PI AF supports time-weighted totalizations that can automatically assign the appropriate units of measure, it is essential that the conversion factor of the UOM be defined to as precise a level as possible for this feature to activate. For example, if you were to define a flow rate UOM such as US gallons per day with a conversion factor of 4.38126E-08, automatic conversions for this UOM would be disabled, whereas if the conversion factor were 4.381263638889E-08, automatic conversions would be active.
Formula	Converts a UOM based on a complex conversion calculation. For more information, see UOM conversion calculation with a formula.

9. Optional. To establish an alternative UOM name so that users can see attribute values returned in their preferred UOM, in the **Mappings** list, select a UOM name from the **Mapping** list for each UOM group.

> **Note:**
> Although default mappings are provided for **Metric** and **US Customary**, you can modify them as needed. To modify multiple UOM mappings in UOM groups, OSIsoft recommends you use PI Builder.

10. Click **OK** to save the new UOM.

UOM conversion calculation with a formula

In general, OSIsoft recommends you specify UOM conversion calculations with a scaling factor and base offset from the Reference UOM value.

> **Caution:**
> Conversions that are based on **Formula** have some limitations as well as impact on performance. You should only select **Formula** when **Simple** conversions are inadequate.

Conversion calculation limitations

The following constraints exist with UOM conversion calculations.

Conversion type	Limitations
Factor	None.
Factor with offset	For delta conversion calculations, you need to define a separate UOM. See the Temperature (Delta) UOM class for examples.
Formula	• Delta calculations are not assigned UOMs. • Summary calculations over a range of data are not assigned UOMs. This affects analysis expression functions, such as Maximum, Minimum, Popular Standard Deviation, Range, Standard Deviation, and Total. It also affects programmatic Summary data calls against PI point data references with a UOM assigned. • Searches for attribute values exclude attributes whose value is measured in formula UOMs.

UOM formula evaluation

PI AF uses C# for UOM formula evaluation. Follow these guidelines:

- Write all Units of Measure in terms of the UOM abbreviation. If an abbreviation is not a valid C# variable name, enclose it in brackets.
- Adhere to C# evaluation rules.
- You can optionally invoke standard .NET static methods, such as **`Math.Log10()`**, to perform the computation. You are limited to what is available in the System Assembly (.NET Framework Math Class (https://msdn.microsoft.com/en-us/library/System.Math(v=vs.110).aspx)).

Formula conversion method example

As an example, `watts` (`W`) is the canonical UOM in the `Power` class. To create a UOM of `Decibel-milliwatts` (`dBm`) in the `Power` class, you could enter the following formula:

```
W = Math.Pow(10,(dBm - 30)/10)
```

```
dBm = 10 * Math.Log10( W ) + 30
```

Calculate conversion values for a UOM

Use the Unit of Measure Conversion Calculator to convert a given UOM quantity into an equivalent value for other members in its class.

Procedure

1. In the navigator, click **Unit of Measure**.
2. In the Unit of Measure browser, select a UOM class.
3. In the **Quantity** field of the **Conversion Calculator** pane, enter the quantity you want to convert.
4. Optional. In the **UOM** field, change the UOM from the default value to another UOM in the class.

   Converted values for each UOM in the class are displayed.

UOM groups

Beginning with PI AF 2017 R2, units of measure that are included as part of PI AF are mapped to UOM groups, such as **Metric** and **US Customary**. Once you have selected which UOM group you prefer to use at your location, you can display all attribute values with units of measure mapped to that UOM group rather than the default UOMs that are defined in attribute templates.

You select which UOM group you prefer in **Tools** > **Options**. When no preferred UOM group is selected, all attribute values are displayed with the default UOM defined in attribute templates, unless **Use Source Unit-of-Measure for attribute display** is also selected. For more information, see PI System Explorer customization options.

Sample mappings for a UOM class

As an example, the table below illustrates mappings for the Length UOM class.

UOM	Metric UOM group	US Customary UOM group
centimeter		inch
foot	meter	
inch	centimeter	
international nautical mile		
kilometer		mile
meter		yard
millimeter		sixteenth of an inch
sixteenth of an inch	millimeter	
yard	meter	

If a mapping is not defined, such as international nautical mile, a UOM is not mapped and uses the original UOM setting.

UOM database

UOM groups are available to all users and are stored on the PI AF server as part of the UOM database. Each user's UOM group selection in **Tools** > **Options** is also stored in the UOM database.

Users with permission to modify the UOM database can modify the UOM groups and mappings, including those shipped by default. For more information, see Configure security for the UOM database.

UOM database import/export

You can import and export UOM groups, as well as their mappings, with **File** > **Export to File** and **File** > **Import from File**. When you export, ensure that **Include All Referenced Objects** option is selected.

Manage UOM groups

In addition to the **Metric** and **US Customary** UOM groups that are installed with PI System Explorer, you can create additional UOM groups to which you can map units of measure. You can also modify and delete group names and descriptions.

Before you start

To create, modify, and delete UOM groups, you must have Write permission for the UOM database.

Procedure

1. To manage UOM groups, click **Unit of Measure** in the navigator.
2. Choose from the following actions.
   - On the toolbar, click the **UOM Groups** icon.
   - Click **File** > **UOM Groups**.
3. Choose from the following actions.

To...	Do this...
Create a new group	a. Click **New UOM Group**. b. In the Unit-of-Measure Properties window, enter a unique name for the UOM group in the **Name** field. c. Optional. In the **Description** field, enter a description for the UOM group. d. Click **OK**. The new group is added to the list in the UOM Groups window.
Modify a group	a. Select the group you want to modify and click **Properties**. b. In the Unit-of-Measure Properties window, make changes as needed to the name and/or description. c. Click **OK**.

Delete a group	a. Select the group you want to delete and click **Delete**. b. In the Delete confirmation window, click **Yes**. **Note:** If the deleted UOM group was previously selected in **Tools** > **Options** on the **General** tab, the **Display UOM Group** value is reset to **<None>**. Attributes are reset to the default UOM defined in the associated element template.

4. Check in your changes to the UOM database.

Origin of UOMs

Beginning with PI AF 2017 R2, you can check the origin of a UOM definition in the UOM database. The **Origin** field is included in the Unit of Measure Properties window, as well as the UOM viewer. It is also supported in PI Builder.

Possible values shown in the **Origin** column are:

Origin	Description
`Unknown`	The origin of how the UOM was defined is unknown.
`System Defined`	The UOM is system-defined and has not been modified. **Note:** If only a UOM mapping is changed, **Origin** remains set to `System Defined`.
`System Modified`	The UOM was system-defined but the **Name** and/or **Description** fields have been modified.
`System Replaced`	The UOM retains its original system-defined abbreviation and canonical UOM values but other fields besides **Name** and **Description** have been modified.
`User Defined`	The UOM was created by the user.

If a UOM is modified that results in **Origin** being set to either `System Modified` or `System Replaced`, but is later reset to the original definition, **Origin** also reverts to `System Defined`.

Security configuration in PI AF

PI AF 2015 (version 2.7) and later uses a security model that is similar to that which PI Data Archive uses. This model relies on integrated Windows security for authentication, but provides its own authorization to PI AF objects using PI AF identities and mappings.

> **Note:**
> The security configuration information described in this section presumes a PI AF 2015 server and a PI AF 2015 client.

The PI AF security model enables administrators to configure access for PI AF identities at each level of the PI AF hierarchy. PI AF uses Windows integrated security to grant or deny connection to the PI AF server, to view or edit databases, and to change collections.

For detailed information on how security is implemented on PI Data Archive, refer to the *PI Data Archive Security Configuration Guide* or see the Security configuration topic "Introduction to PI Data Archive security" in Live Library (https://livelibrary.osisoft.com).

Video

For information on how to create, map, and grant permissions to PI AF identities, watch this video:

https://www.youtube.com/watch?v=bWqHvoLuXzs&rel=0

Topics in this section

- PI AF identities and mappings
- Built-in PI AF identities
- Security hierarchy
- Security inheritance of PI AF objects
- PI AF access rights
- Deny permission option
- Security Configuration window
- PI AF server security
- PI AF database security
- PI AF collection security
- PI AF object security
- Configure security for the UOM database
- Differences from PI Data Archive security model

PI AF identities and mappings

A PI AF identity represents a set of access permissions on the PI AF server. Each PI AF mapping points from a Windows user or group to a PI AF identity. Beginning with PI AF Server 2015 (v2.7), you cannot directly grant a Windows user or group access to a PI AF server resource (such as an element collection or objects). Instead, you create a PI AF identity that has that

access and then you create a PI AF mapping between the Windows user or group and that PI AF identity.

Members of the Windows groups that are mapped to a PI AF identity are automatically granted the access permissions for that PI AF identity. For example, in the following illustration, the PI AF identity called `Engineers` has read/write access to the Elements collection. Because the Active Directory (AD) group `Engineering Team` is mapped to `Engineers`, all the members in that AD group get read/write permission for the Elements collection.

AD group mapping to a PI AF identity

Multiple identities

A single Windows user can be mapped to multiple PI AF identities, typically via mappings of the various Windows group memberships to which he or she belongs. A user is granted permissions based on *all* the PI AF identities to which he or she is mapped. Effective permissions are determined by taking the union of all identities' allowed permissions and removing the union of all denied permissions. For example, in the following illustration, the Windows user Bob belongs to both AD groups. Bob therefore gets the permissions that are configured for `PI AF IDENTITY`$_1$ and `PI AF IDENTITY`$_2$.

Windows user with cumulative access permissions

Additionally, a user must have read permission on a PI AF database to be able to read any object within it. Likewise, a user must have write permission on a PI AF database to write to any object within it.

For more information on working with identities and mappings, see Manage identities and Manage mappings.

Built-in PI AF identities

PI AF includes the following built-in PI AF identities:

PI AF identity	Description
`Administrators`	By default, this identity has all access permissions to every collection and object on the PI AF server, including all databases. It cannot be modified or deleted. Mappings, however, can be added and removed, and this identity can be denied access permissions to objects if the need arises. OSIsoft recommends that access to this identity be restricted to only a few users.
`Asset Analytics`	(Part of PI Analysis Service installation.) This identity has the necessary access permission to work with analyses. By default, the account used to run PI Analysis Service is mapped to this identity during installation. Mappings to this account can be added or removed.
`Asset Analytics Recalculation`	(Part of PI Analysis Service installation.) This identity has Execute permission, allowing users mapped to it to backfill and recalculate analyses.[1]

PI AF identity	Description
`Engineers`	This identity has the same privileges as `Administrators`, with the exception of the **Admin** (a) permission. This identity is also not allowed to delete PI AF databases. OSIsoft recommends that this identity be restricted to those users who are defining the asset database. Additional identities should be created to narrow the scope of access within PI AF.
`Notifications`	(Part of PI Notifications Service installation.) This identity has the necessary access permission to work with notification rules. By default, the account used to run PI Notifications Service is mapped to this identity during installation. Mappings to this account can be added or removed.
`Owner`	This read-only identity can be explicitly added to the security configuration of specific PI AF objects to enable administrator users to configure privileges for the owner of an object. The following restrictions apply: • You cannot add mappings to this identity. • You cannot modify, disable, or delete this identity. • You cannot remove or deny Read permission for this identity. The owner of an object always has Read and Read Data permissions on that object.
`World`	This identity has read access permissions to every collection and object on the PI AF server. It cannot be modified or deleted. Mappings, however, can be added and removed. By default, this identity is mapped to the Windows `Everyone` users group.

[1] If you have installed PI AF 2017 (version 2.9) or later versions for the first time, users need to be added to the `Asset Analytics Recalculation` identity in order to backfill or recalculate. If you are upgrading from versions prior to PI AF 2017 (version 2.9), all users will automatically be mapped to `Asset Analytics Recalculation` identity. It is recommended that upon upgrading, such automatic mapping for all users is removed and users that require backfilling or recalculation permissions are explicitly mapped.

Security hierarchy

PI AF uses Windows integrated security to authenticate users and establish their PI AF identities through mappings. PI AF uses the PI AF identities to control read, write, delete, and various other permissions on PI AF components shown in the following illustration. Each securable PI AF object (element, event frame, and notification, and so on) throughout the hierarchy has an associated security descriptor that contains the access permissions information for that object.

All PI AF objects of the same type belong to a collection. For example, every PI AF element in a database belongs to the Elements collection for that database. Each collection also has an associated security descriptor that contains access permission information. Security descriptors for some collections are configured for an entire PI AF server (such as Identities and Mappings), whereas others (such as Analyses, Elements, and Event Frames) can be configured for a specific database.

PI AF hierarchy of securable collections

For more information on collection security, see PI AF collection security.

Security inheritance of PI AF objects

The following table provides details on the security that is required to create specific PI AF objects. Note that the Owner of an object always has Read and Read Data permissions on that object.

Object	Security inheritance
Element	A child element inherits the security of its parent element if it is added as a Parent-Child (strong) or Composition reference. For more information, see Permission inheritance of element objects. The security rights of an element created at the PI AF SQL Server database level are derived from the Element collection associated with the PI AF SQL Server database. For more information, see PI AF collection security.
Event frame	A child event frame inherits the security of its parent event frame if it is added as a Parent-Child (strong) reference. If an event frame does not have a Parent-Child reference, security rights are derived from the element template if it was created from a template. If it was not created from a template, security rights are derived from the Event Frame collection associated with the PI AF SQL Server database.
Transfer	If a transfer is created from a template, security rights are derived from the template. If it was not created from a template, security rights are derived from the Transfer collection associated with the PI AF SQL Server database.
Case	Security rights are derived from the analysis that owns the case.

For all other types of objects, initial security access rights are calculated from the corresponding collection associated with the PI AF SQL Server database. For example, security access rights for a notification are initially derived from the PI AF SQL Server database's Notification Rule collection.

Note:
When an analysis or analysis template is associated with a legacy notification or notification template, (Notifications 2012 or earlier), security access rights for the two objects are synchronized.

Permission inheritance of element objects

When you change access permissions for an element, the access permissions for any parent or child elements might also change. The behavior depends on the reference type.

Parent-child reference type

When an object or collection is created, a default set of access permissions is assigned, based on the access permissions that are set on the parent. When access permissions are set on a parent, the following **Child Permission** settings in the Security Configuration window are evaluated:

Option	Description
Do not modify child permissions	Prevents access permissions that have been set for the current object or collection from being replicated to child collections and objects in the PI AF hierarchy. This option is the default when the connected PI AF server is running 2.5 and earlier versions.

Option	Description
Update child permissions for modified identities	For each selected item on the **Items to Configure** list in the Security Configuration window, replicates the access permissions for all child collections and objects for each identity on the **Identities** list whose access permissions have been modified. This option is the default when the connected PI AF server is running 2.6 and later versions. This option is not available when the connected PI AF server is running 2.5 and earlier versions.
Replace child permissions for all identities	For each selected item on the **Items to Configure** list in the Security Configuration window, replaces all child permissions for every identity on the **Identities** list with the parent access permissions. **Tip:** Before you apply this option, OSIsoft recommends that you review access permission settings for all items on the **Items to Configure** list to avoid unintentionally overwriting custom permissions that may have been applied elsewhere in the collection hierarchy. Review the following example for clarification.

Examples

The following hierarchy has three elements, each with a different access permissions setting.

Sample element hierarchy

- The `Administrators` and `World` identities have access permissions to all three elements: `EasternUS`, `SavannahSite`, and `ProductionLine1`.
- The `Savannah_IT` identity has access to the `SavannahSite` element.
- The `SavnEngineers` identity has access to the `ProductionLine1` element.

Access permissions for sample hierarchy

Suppose you want to add a `CorporateEngineering` identity to each element in the hierarchy. You would add the identity to the parent element `EasternUS`:

Add identity to parent element

To replicate the parent permissions without affecting identities already assigned access permissions, you would set the **Child Permissions** option to **Update child permissions for modified entries**. The security string for all three elements shows that the `CorporateEngineering` identity has been added, but the other identities remain unchanged:

Replicate identity and modify existing access permissions

To replicate the parent permissions for all identities down the hierarchy, you would set the **Child Permissions** option to **Replace child permissions for all identities**. The security string for all three elements shows that the `CorporateEngineering` identity has been added, but has replaced the access permissions with those assigned to the `EasternUS` element:

Replicate identity and replace existing access permissions

Notice that the `Savannah_IT` and `SavnEngineers` identities no longer appear in the security string for the `SavannahSite` and `ProductionLine1` elements because they were not included in the **Identities** list of the parent `EasternUS` element.

Composition reference type

Access permissions for child and parent are always the same.

If you change the access permissions for the child, the parent access permissions are automatically changed to match the child permissions. Similarly, if you change the access permissions for the parent, the child access permissions are automatically changed to match the parent permissions. These changes cascade down (and up) through the hierarchy.

Weak reference type

Access permissions are never inherited.

PI AF access rights

The following table describes the access permissions you can assign to PI AF identities for all objects in the PI AF hierarchy.

Access right	Security string abbreviation	Definition
`Read`	`r`	Enables a user to view the object. `Read` security rights are required to view the object in client applications. The following objects always have `Read` permission regardless of their security settings, so long as the logged-in user has Read access to the PI System and the PI AF SQL Server database. The one exception is the notification contact template, where the logged-in user only requires Read access to the PI System: • Analysis template • Categories • Element template • Enumeration set • Event frame template • Model template • Notification contact template • Notification template • Reference type • Transfer template • UOM database
`Write`	`w`	Enables a user to create and modify an object. The exception is that event frames and transfers also require `Write Data` permission on the element template from which they are created, and cases require `Write Data` permission on the analysis in which they are contained.
`Read/Write`		Enables a user to read and write to the associated object. When selected, automatically selects the `Read` and `Write` permissions.
`Read Data`	`rd`	Enables a user to read non-configuration values from attributes of elements (the **Configuration Item** property for an attribute is cleared). Additionally, this permission controls whether a user can see transfers created from a specific transfer element template. Similarly, it controls whether a user can see cases created in a specific analysis. If the following objects have `Read Data` permission, they are also granted `Read` permission: • Case • Element • Event frame • Model • Notification • Transfer
`Write Data`	`wd`	Enables a user to write non-configuration values to attributes of elements (the **Configuration Item** property for an attribute is cleared). Additionally, this permission controls whether a user can create or modify event frames or transfers created from a specific transfer element template. Similarly, it controls whether a user can create or modify cases in a specific analysis.
`Read/Write Data`		Enables a user to read data and write data to the associated object. When selected, automatically selects the `Read` and `Write Data` permissions.

Access right	Security string abbreviation	Definition
`Delete`	`d`	Enables a user to delete an object. Delete security rights are required to delete an object, either directly or indirectly by removing it from other objects.
`Execute`	`x`	Enables a user to queue backfilling or recalculation of analyses in the analysis service. It also enables a user to perform most actions on an analysis case.
`Admin`	`a`	Enables a user to modify the security settings, or owner, of an object. Administration security rights are required to force an **Undo Check Out** on an object that is checked out to another user, as well as to lock and unlock an event frame. **Note:** Users with the administration permission on the PI AF server object are granted all rights not only to the system, but to all objects within the system, including databases.
`Subscribe`	`s`	Enables a user to subscribe and unsubscribe to a notification.
`Subscribe Others`	`so`	Enables a user to subscribe and unsubscribe other users to a notification.
`Annotate`	`an`	Enables a user to annotate and acknowledge event frames. **Note:** This access right was added in PI AF 2016. After an upgrade from earlier server versions, objects with the `Write Data` (`wd`) access right are granted the `Annotate` access right automatically. Both client and server upgrades must use this new permission.

PI Data Archive permissions

In order to connect to a PI Data Archive server, PI AF requires that `Read` permission be configured on that server. Note that because determining the permissions on a PI Data Archive server requires an actual connection to be made, the full list of PI Data Archive servers configured on a client machine is always available. PI Data Archive also always has `Delete` permission.

Deny permission option

You select the **Deny** option for these cases:

- To exclude a subset of an identity mapping that has allowed permissions.
- To exclude one special permission when you have already granted full control to an identity.

Note:
PI Module Database does not support the Deny option. If you are using both PI MDB and PI AF, avoid the Deny option to prevent synchronization problems.

Security Configuration window

You can open the Security Configuration window anywhere in PI System Explorer where a **Security** link is displayed in a window or **Security** is displayed on a context menu.

Security Configuration

Items to Configure list

The content of the **Items to Configure** list is determined by the hierarchy context and indicates whether you can configure access permissions for the entire PI AF hierarchy, a single database and containers, or a single container and single object.

To configure access permissions for only some of the items, you can clear those you do not want to configure.

Identities list

The **Identities** list contains a list of identities that have permissions for *all* checked items in the **Items to configure** list. You use the **Add** and **Remove** functions to manage which identities appear on the list.

If the currently connected PI AF server does not support identities (version 2.6 or earlier), the Security Configuration window displays access permissions for Groups or Users and you can add and remove Windows accounts using standard Active Directory windows.

Permissions list

As each identity is selected in the **Identities** list, the permissions associated with that identity for the checked entries in the **Items to Configure** list are shown in the **Permissions** list. You can allow or deny permissions as desired for each identity without losing the changes. The access permissions you set for each identity are retained until you click the **OK**, **Cancel**, or **Apply** button.

Note:
If more than one **Items to Configure** entry is selected and the currently selected identity has one of the permissions in one or more, but not *all* entries selected in the **Items** list, the checkbox for that permission displays a dot (■) to indicate some entries in the **Items** list have the permission set whereas some do not.

PI AF server security

When PI AF Server is first installed, admin access (all permissions) is given to the built-in `Administrators` identity, and read access to all objects and collections is given to the `World` identity. For all other identities that are mapped to Windows users and groups, an administrator needs to configure appropriate access to the PI AF server, databases, collections and objects.

Configure security for a PI AF server

Configure access permissions for a connected PI AF server so that a PI AF identity can access databases, collections and objects.

Before you start

To connect to a PI AF server, an identity must have read permissions to the PI AF server object.

Procedure

1. Choose one of the following methods to open the Security Configuration window for a PI AF server:

To open from ...	Do this ...
Servers window	a. Select **File** > **Connections**. b. In the Servers window, right-click the default connected ( ) PI AF server and click **Security**.
PI AF Server Properties window	a. On the toolbar, click . b. In the PI AF Server Properties window, click the **Security** link.
Select Database window	a. On the toolbar, click the **Database** button. b. In the Select Database window, click the **Edit Security** button.

   In the **Items to Configure** list of the Security Configuration window, the server and every collection is selected.

2. Clear those collections that you do not want to be assigned the same permission settings as the server. For example, you might want to assign access rights for server-wide collections such as Mappings and Identities, but use a different set of access permissions for databases and their collections.

3. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set.

Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

4. Select one of the **Child Permissions** options. For more information, see Permission inheritance of element objects.
5. Click **OK**.

PI AF database security

To view a database, an identity must have at least read permissions to the database object. You configure permissions individually for each database, or for the entire databases collection.

Beginning with PI AF 2017 R2, it is no longer necessary for users to have write permission on a PI AF database in order to have write permission on other objects in the database. Users need only be assigned an identity with write access to objects such as elements to be able to edit them. To prevent users from having write access to specific objects, assign them to an identity with read-only access to those objects.

Topics in this section

- Configure security for new PI AF databases
- Configure security for a single PI AF database

Configure security for new PI AF databases

Configure default PI AF access rights for all identities that need access to databases on a PI AF server.

Procedure

1. On the toolbar, click the **Database** button.
2. In the Select Database window, click the **Edit Security** button.

   In the **Items to Configure** list of the Security Configuration window, the server and every collection is selected.
3. In the **Items to Configure** list, clear the server and the following collections: **Contacts**, **Notification Contact Templates**, **Identities**, and **Mappings**. **Databases** and all **Database - *Collections*** should be selected.

4. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check box beside each permission you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are checked). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

5. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
6. Click **OK**.

Configure security for a single PI AF database

For a single database, configure PI AF access rights for identities that need access to its collections and objects.

Procedure

1. On the toolbar, click the **Database** button.
2. In the Select Database window, right-click a database in the **Databases** list and click **Security**.

   In the **Items to Configure** list of the Security Configuration window, the database and every collection in the database is selected.
3. Optional. Clear any collection that you do not wish to modify from their default settings and assigned identities.
4. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are checked). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see PI AF identities and mappings.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

5. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
6. Click **OK**.

PI AF collection security

The security descriptor for a PI AF collection determines access permissions to that collection, as well as default access permissions for new objects in the collection. For example, the Elements collection permissions determine which identities can create and delete elements. The Elements collection permissions are replicated as the security descriptor for any newly-created Element objects.

You can configure access permissions to collections at several points in the PI AF hierarchy. You can set them at the server level so that permissions assigned to identities on the server are also assigned to the same identities for every collection in every database (see Configure security for a PI AF server). You can set them at the database level so that permissions assigned to identities in a database are also assigned to the same identities for every collection in that database (see Configure security for a single PI AF database).

Note:
Library objects (Templates, Enumeration Sets, Reference Types, Tables, Table Connections, and Categories) always have Read (r) permission regardless of their security settings.

Configure security for collections

Assign access permissions for a specific collection in a PI AF database.

Procedure

1. Choose one of the following methods to open the Security Configuration window for a database collection:

To open ...	Do this ...
Elements collection	In the Elements browser, right-click the Elements collection ( ) and click **Security**.
Event Frame collection	In the Event Frames browser, right-click an event frame search ( ) and click **Security**.
Transfer collection	In the Event Frames browser, right-click a transfer search ( ) and click **Security**.
Element Templates collection	In the Library browser, right-click **Element Templates** and click **Security**. **Note:** Access permissions for the Element Templates collection also set access permissions for Event Frame Templates, Model Templates, and Transfer Templates.
Enumeration Sets collection	In the Library browser, right-click **Enumeration Sets** and click **Security**.
Reference Types collection	In the Library browser, right-click **Reference Types** and click **Security**.
Tables collection	In the Library browser, right-click **Tables** and click **Security**.
Table Connections collection	In the Library browser, right-click **Table Connections** and click **Security**.
Categories collection	In the Library browser, right-click **Categories** and click **Security**.

In the **Items to Configure** list of the Security Configuration window, the collection is selected.

2. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as needed.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check box beside each permission you wish to set. c. Repeat for each identity.

Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

PI AF object security

You can set specific access permissions for an identity that differ from the default settings inherited from elsewhere in the PI AF hierarchy on any object (or object group) and collection in a database.

Topics in this section

- Configure security for objects
- Configure security for analyses and analysis templates

Configure security for objects

Set access permissions for identities to objects in the browser or the viewer in situations where access needs to be different from inherited permissions. You can also set custom permissions for identities to a specific collection in the Library.

Procedure

1. Choose from the following actions:

To set access permissions for ...	Do this ...
A browser object	Right-click the object and click **Security**.
Multiple objects	a. With multiple objects listed in the viewer (after a search, for example), use standard Windows keystrokes to highlight contiguous (SHIFT+<click>) or non-contiguous (CTRL +<click>) objects. b. Right-click and click **Security**.

2. In the **Identities** list, review the identities and validate their permissions. You can also add to and remove an identity from the list.

To ...	Do this ...
Add or modify permissions of existing identities	a. Select an identity in the **Identities** list. b. In the **Permissions for *Identity*** list, select or clear the **Allow** or **Deny** check boxes beside the permissions you wish to set. c. Repeat for each identity.
Add another identity to the **Identities** list	a. Click **Add**. b. In the Select Identity window, select the identity you wish to add. c. Click **OK**. The identity is added to the list. d. In the **Permissions for *Identity*** list, set the access permissions as needed (the default setting is all are selected). **Tip:** If you need to create a new identity and mapping, click **New Identity**. For more information, see Manage identities.
Remove an identity from the **Identities** list	a. Select an identity in the **Identities** list. b. Click **Remove**.

3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

Configure security for analyses and analysis templates

Use the Security Configuration window to manage access permissions for PI AF identities that can work with analyses or analysis templates.

For information on specific permissions that are needed to work with analyses, see PI AF access rights.

Procedure

1. Choose from the following actions:

To ...	Do this ...
Configure access permissions for an analysis	a. In the Elements browser, select the element that contains the analysis. b. Click the **Analyses** tab. c. Right-click the analysis name and click **Security**.
Configure access permissions for an analysis template	a. In the Library browser, select the element template that contains the analysis template. b. Click the **Analysis Templates** tab. c. Right-click the analysis name and click **Security**.

Configure access permissions for all analyses in a PI AF database	a. On the toolbar, click **Database**. b. In the **Select Database** window, right-click the database name and click **Security**. c. In the Security Configuration window, clear the **Item** check box above the **Items to Configure** list. d. Select the ***database* - Analyses** item.
Configure access permissions for all analysis templates in a PI AF database	a. On the toolbar, click **Database**. b. In the **Select Database** window, right-click the database name and click **Security**. c. In the Security Configuration window, clear the **Item** check box above the **Items to Configure** list. d. Select the ***database* - Analysis Templates** item.

2. In the **Identities** list, review the identities and validate their permissions. You can also add to and remove an identity from the list, as described in Configure security for objects.
3. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
4. Click **OK**.

Configure security for the UOM database

You cannot set permissions for individual UOMs or UOM classes. However, you can set permissions for the entire UOM database.

Before you start

UOMs always have the Read permission regardless of other security settings.

Procedure

1. In the navigator, select **Unit of Measure**.
2. On the toolbar, click **UOM Security**.

   In the **Items to Configure** list of the Security Configuration window, the **Unit-of-Measure Database** item is selected.
3. In the **Identities** list, review existing identities and validate their permissions. You can also add and remove identities, as described in Configure security for objects.
4. In the **Child Permissions** section, choose one of the permission inheritance options. For more information, see Permission inheritance of element objects.
5. Click **OK**.

Differences from PI Data Archive security model

Although the security model is similar to that which PI Data Archive uses, there are a number of differences:

- The Deny privilege is supported. See Deny permission option.
- A more expansive set of access permissions is available. See PI AF access rights.
- The equivalent of `PI user` and `PI group` identities are not supported.
- PI trusts and PI explicit logins are not supported.
- The concept of an undeletable flag on an identity or mapping is not supported.
- Mappings cannot be disabled. Only PI AF identities can be disabled.
- A single Windows user identity can be mapped to more than one PI AF identity.
- A different built-in set of identities is installed: `Administrators`, `Engineers`, and `World`.

Event frames in PI AF

Capturing important events in your process and collecting relevant data around those events can help analyze why they occurred. For example, you can closely monitor events such as asset downtime, process excursions, equipment startup or shutdown, and environmental excursions to identify possible causes or potential points of failure.

Collecting data around repeatable time periods such as product tracking batches, product runs, or operator shifts can help make those processes more efficient. The capture of comprehensive data associated with such an event helps track, compare, or analyse the process or event.

Just as elements allow you to collect and store data about assets, event frames allow you to collect and store data about events. OSIsoft recommends you use asset analytics to track your events using event frames. PI Datalink, PI Vision, and PI WebParts are the client tools that support event frame visualization.

An event frame encapsulates the time period of the event with relevant, comprehensive asset data:

Sample event data

Name	Start Time	End Time	Primary Element	| Operator	| Phase	| Reason Code	| Level.Start	| Temp.Max	| Temp.Start
Template: Downtime									
DT20120805 22:56	8/5/2012 10:56:00 PM	8/5/2012 11:00:00 PM	RE200	Vivian White	Material Add 1	DRIVE FAILURE	0 L	12.935262298...	0.41429677605629 deg C
DT20120805 23:00	8/5/2012 11:00:00 PM	8/6/2012 12:00:00 AM	RE400	Don Wallace	XFER_OUT	No Oil	69.978393554...	110.66223907...	40.0944023132324 deg C
DT20120805 23:26	8/5/2012 11:26:00 PM	8/5/2012 11:29:00 PM	RE100	Bobby Johnson	Material Add 1	No Oil	4.6750192642...	5.4764796495...	3.44453287124634 deg C
DT20120805 23:56	8/5/2012 11:56:00 PM	8/6/2012 12:00:00 AM	RE200	Vivian White	Inactive	DRIVE FAILURE	3.5982928276...	8.9376313209...	8.93763160705566 deg C
DT20120806 0:14	8/6/2012 12:14:00 AM	8/6/2012 12:18:00 AM	RE400	Don Wallace	XFER_OUT	ESTOP	74.283164978...	56.506531524...	56.5065307617188 deg C
DT20120806 0:26	8/6/2012 12:26:00 AM	8/6/2012 12:29:00 AM	RE100	Bobby Johnson	Dwell	UNKNOWN	62.131874084...	101.77695465...	98.174674987793 deg C
DT20120806 0:56	8/6/2012 12:56:00 AM	8/6/2012 1:00:00 AM	RE200	Vivian White	Agitate	ESTOP	46.672977447...	62.632131067...	62.6321296691895 deg C
DT20120806 1:14	8/6/2012 1:14:00 AM	8/6/2012 1:18:00 AM	RE400	Don Wallace	Heat	DRIVE FAILURE	68.781196594...	96.730087280...	72.2844924926758 deg C

Videos

For information on event frames, watch this video:

https://www.youtube.com/watch?v=4XPL7J7u61U&rel=0

Alternatively, you can view all the videos on the event frames playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2JbcunItGs7JS6kFl3WYMb-p3D&rel=0

Topics in this section

- Structure of event frames
- Advantages of event frames
- Examples of event frames
- Event Frames or Batch?
- Ways to create event frames
- Visualization of event frames
- Event frame templates

- Create event frames
- Value capture for event frames and transfers
- Lock event frames or transfers
- Acknowledgement of event frames
- Transfers

Structure of event frames

Each event frame has a name, start time, end time, one or more attributes, and one or more referenced PI AF elements. As with elements, you should create event frame templates to standardize and manage the attributes for different types of events.

With event frames you can easily search the PI System for the events themselves, rather than search for events by time. You can configure event frames to return all related event data automatically in real-time so that you do not need to query multiple systems for event and process data, and then merge them together manually. You can also set up event frames to retrieve historical data.

When to use event frames

There are two categories of trackable events that would fit an event frame profile:

- "Good" events: Events that you want to track as a normal part of business, such as product tracking, shifts, and so on.
- "Bad" events: Events that are unexpected and need to be analyzed and perhaps fixed quickly if they ever occur, such as expected shutdowns or excursions. These are events that you want to track and report in aggregate, over a period of time.

Asking questions such as these can help identify events or conditions that must be tracked:

- What are all the times that event X occurred on this type of asset?
- Can I associate data from different tags for a time-range, or for a single point in time?
- What is the associated data for a particular time period when a problem occurred or may occur in the future?
- What are the critical process events that someone needs to be notified on?
- Are there digital states for PI system tags that are significant when they change, and must they trigger other actions?

Advantages of event frames

Event frames provide these advantages:

- **Flexibility**

  Event frames:

  - Reference multiple elements within the same event.
  - Support multiple overlapping events on a PI AF element.
  - Capture any event; a "batch" is just one type of capturable event.

- **Easy search options**

  Searches can be specified on a number of configurable event frame attributes: it is optional to specify a time range. For example, you can search just by entering the name of the event frame. The most commonly searched event frame attributes can be configured as indexed attributes through event frame templates, which speeds up end-user searches.

- **Scalability**

  Event frames are extremely scalable whereas search performance degrades with a large number (tens of thousands) of batches.

Examples of event frames

Example of event frames in a wind power generation company

Take the example of a wind power generation company that has different types of windmills across different locations and uses PI AF to organize their data. Their asset framework structure is based off a base element template of type "windmill", and has various child templates based off OEM, model, and megawatt ratings. The wind operator may want to win favorable warranty contracts by showing that the blades are operating safely. Or, the operator may need to monitor parameters such as oil bearing pressure, maximum voltage, power-factor, or performance of electronic and mechanical brakes in the minutes before process trips occur. In all these cases, you can associate event frames with the specific event frame attributes.

The sample windmill element may include these asset attributes and associated event frame attributes:

Asset attribute	Event frame attribute
RPM	Instantaneous and maximum speeds
Voltage	Maximum voltage
Power	Power at current time
Yaw	Yaw at current time
Pitch	Pitch at current time
Oil bearing pressure	Maximum oil bearing pressure

Since all types of turbines share many identical attributes, you can create just one event frame template and use it to monitor similar events across the different assets. For example, you may be interested in the RPM attribute to capture a speed-based event. Using the special notation `.\Elements[.]|RPM` in the template enables you to use the template on any windmill, and access the attribute of the particular referenced element.

Example of event frames using Asset Analytics

For an example of creating event frames using asset analytics, see Create event frames automatically to track inefficiency.

Event Frames or Batch?

Batch provides you with a means to generate batch event data based on PI tags. Event Frames provide you with a way to track and analyse process and business data related to events that are defined on PI AF attributes.

Different companies and users come from different levels of PI System implementation and use of PI System features; hence, the recommendation of whether to choose Batch or Event Frames is highly subjective. You may choose Batch or Event Frames depending on your process and business needs, such as:

- You are a Batch user and use RtReports.

  You must continue to use Batch. Typical Batch use-cases include paper grade runs, chemical batch processing operations, and so on.

- You do not currently use Batch but have batch use-cases, or must use RtReports.

  You must use Batch. OSIsoft will provide a migration path in the future when the PI System begins to support these functions.

- You do not use Batch and do not have batch use-cases.

  You must use Event Frames. Typical Event Frame use-cases include downtime or outages in power industries, excursions in water utility industries, and so on.

> **Note:**
> OSIsoft does not recommend that you run dual or parallel instances of Batch and Event Frame interfaces in production as it adds to your migration complexity.

Unsupported PI System functionality with Event Frames

- RtReports - PI AF and Event Frames are not currently supported by RtReports; however, you can use PI System access products like PI OLEDB Enterprise to write SQL queries for third-party reporting solutions.

Ways to create event frames

Although you can create event frames with several different OSIsoft products, OSIsoft strongly recommends that you use asset analytics.

Asset analytics

PI Data Archive supports creation of single-layer events and can trigger expressions that open or close event frames. See Event frame generation analyses.

PI Event Frame Generator (PI EFGen)

PI EFGen uses both PI SDK and AF SDK to create a hierarchy of events or to convert a PI BaGen structure to an event frame generator structure. Use PI System Explorer or PI Builder to create the event frame templates, associated attributes and PI point references.

For instructions on how to create event frames with PI EFGen, watch this video:

https://www.youtube.com/watch?v=UbideDwhawg&rel=0

PI interfaces for batch and manufacturing execution systems

PI Batch Framework 3.x and later versions can create event frames within a PI AF database or PI Batch objects within a PI Batch database. For more information on creating event frames and using PI interfaces to populate the PI AF database with events-based data, see:

- "PI interfaces for batch and manufacturing execution systems" in Live Library (https://livelibrary.osisoft.com)
- "PI Event Frames Interface Manager" in Live Library (https://livelibrary.osisoft.com)

Programmable interfaces

You can create your own custom program using AF SDK and PI ACE to create and monitor events.

Manual creation of event frames

OSIsoft strongly discourages manual creation and management of event frames and encourages you to use PI Data Archive with asset analytics support or PI Batch, depending on your process needs. See Event Frames or Batch?

Visualization of event frames

You can use PI Datalink, PI Vision, and PI WebParts client tools to visualize event frames. You can also use PI OLEDB Enterprise, PI JDBC Driver, or PI Web Services to integrate event frame data into other third-party reporting tools.

> **Note:**
> PI ProcessBook, PI BatchView, and PI Manual Logger do not currently support event frame visualization.

- **PI Datalink**

  PI Datalink support for event frames includes exploring and comparing hierarchical events. For more information, see "Events in worksheets" in Live Library (https://livelibrary.osisoft.com).

- **PI Vision**

  PI Vision enables you to view and analyze your PI System data during the time range of a particular event. For example, you may want to examine the performance of an asset during an operator shift or compare the data for several assets during a downtime period.
  PI Vision supports event frames through the "Related Events" viewer. For more information, see "Analyzing and comparing events" in Live Library (https://livelibrary.osisoft.com).

- **PI WebParts**

  PI WebParts does not include specific features related to event frame visualization. However, you can create a data set based on a relational data source that retrieves event frame data via PI OLEDB Enterprise. This event frame data can then be displayed in a PI Table web part, for example, and used to provide context, such as start and end times, to other web parts on the page.

Event frame templates

Using event frame templates, you can define and standardize the related data (event frame attributes) associated with different types of events. You can use event frame attributes to provide additional context around the event that are useful for searches and reports. For example, downtime events often have a reason code that users want to search for or filter on during analysis of their downtime events.

You can also configure event frame attributes to reference process data in the context of the event. For example, a temperature excursion event is likely to have an attribute for the maximum temperature. You can configure event frames to record those values for you. Additionally, for each event type, you can configure an index for the most-searched attributes: this enables faster and easier searches on the PI AF server when you track several event types or millions of events.

Data references in event frame templates

When you create event frames dynamically with event frame templates, you typically need to reference elements, attributes, templates, or other related objects based on those event frame templates. Because each individual event frame occurs in a slightly different context, you can use substitution parameters to reference other PI AF objects dynamically, rather than static, absolute references. For example, you can use a downtime event frame template for any number of assets, such as a pump, motor, boiler, compressor, and so on.

In PI point data references, substitution parameters are useful for:

- Building tag name references based on hierarchy or other attribute values.
- Referencing other PI point attributes, such as from an event frame to the primary referenced element.

The main limitation of PI point data references is that the element attribute must also be a PI point data reference.

In String Builder data references, substitution parameters are useful for:

- Obtaining numeric values or string values when referenced element attributes do not have a PI point data reference.
- Creating a string by concatenating multiple values from referenced element attributes and other text.
- Passing a time context, including the end time of the event frame.
- Obtaining the name of referenced elements or their parents.

The limitation of String Builder data references is that they do not perform unit of measure conversions or perform aggregations or calculations.

A specific syntax is required to reference attributes from event frame templates, as described in Data references to attributes in the primary referenced element and Data references to attributes in other elements.

Video

For information on how to set up event frame templates, watch this video:

https://www.youtube.com/watch?v=0gKaw9p4FA0&rel=0

Primary referenced element

Each event frame references one or more elements. The event frame's **Referenced Elements** tab lists all elements that the event frame refers to. The main element referenced by the event frame is called the primary referenced element, and is indicated by a checkmark on the element icon:

By default, the first referenced element added is set as the primary referenced element, but you can change it by right-clicking on another element and selecting **Set as Primary Element Reference**.

> **Note:**
> If you delete the reference to the primary element, PI System Explorer does not automatically set a new primary referenced element. You must select a new primary referenced element manually.

Data references to attributes in the primary referenced element

In data references from an event frame template, you can refer to an attribute in an event frame's primary referenced element with the syntax:

```
.\Elements [.]|AttributeName
```

`.`	Indicates the current object, which is the event frame.
`\Elements`	Specifies the elements collection of the current object.
`[.]`	Represents the default object of the parent object, which is the primary referenced element.
`|AttributeName`	Specifies the name of the attribute.

For example, `.\Elements [.]|Flowrate` obtains the value of the `Flowrate` attribute from the primary referenced element.

Data references to attributes in other elements

In data references from an event frame template, you can reference other elements and attributes with the collection filters shown in the following table.

Collection filter	Example	Explanation
`@Category`	`.\Elements[@Category=MyCategory]`	Obtains the value of the first referenced element that is assigned to the `MyCategory` category.
`@Description`	`.\Elements[@Description=My Description]`	Obtains the value of the first referenced element that has the `My Description` description.
`@Index`	`.\Elements[@Index=3]`	Obtains the value of the third referenced element.
	`.\Elements[@Index=2]|FlowRate`	Obtains the value of the `FlowRate` attribute from the second referenced element.

Collection filter	Example	Explanation
@Name **Note:** You can use wild cards with **@Name**.	`.\Elements[@Name=Tank1]`	Obtains the value of the element named `Tank1`.
	`.\Elements[@Name=*2]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name ends in 2.
@ReferenceType	`.\Elements[@ReferenceType=Area]`	Obtains the value of the first referenced element that uses the `Area` reference type.
@Template	`.\Elements[@Template=MyTemplate]`	Obtains the value of the first referenced element that uses the `MyTemplate` template.
	`.\Elements[@Template=Pump Template]|%attribute%`	Obtains the value of the attribute that has the same name as this event frame attribute from the first referenced element that uses the `Pump Template`.
@Trait	`.\Elements[@Name=Tank*]|Attributes[@Trait=LoLo]`	Obtains the value of the attribute with a `LoLo` trait in the first referenced element whose name begins with `Tank`.
@Type	`.\Elements[@Name=Tank*]|Attributes[@Type=System.Int32]`	Obtains the value of the first attribute that has an `Int32` value type in the first referenced element whose name begins with `Tank`.
@UOM	`.\Elements[@Name=Tank*]|[@UOM=meter]`	Obtains the value of the first attribute that uses the `meter` unit of measure in the first referenced element whose name begins with `Tank`.

You can also combine collection filter criteria, as shown in the following examples:

Example	Explanation
`.\Elements[@Name=*2][@Category=Pump]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name ends in 2 and whose element category is `Pump`.
`.\Elements[@Name=Pump*][@Template=Pump Template]|FlowRate`	Obtains the value of the `FlowRate` attribute from the first referenced element whose name starts with `Pump` and whose element template is `Pump Template`.
`.\EventFrames[@Template=Machine Template]|Pressure`	Obtains the value of the `Pressure` attribute from the first child event frame whose template is `Machine Template`.

Example	Explanation

`.\EventFrames[@Template=Machine Template][@Index=-1]|Pressure`
or
`.\EventFrames[@Template=Machine Template][-1]|Pressure`
	Obtains the value of the `Pressure` attribute from the most recent child event frame whose template is `Machine Template`.
	Note:
	Beginning with PI AF 2017, you can use a negative number for an index from the end of a collection (for example -1 indicates the last item, -2 indicates the penultimate item).
	The **`@Index`** filter is optional if another filter is specified before the index filter.

Create event frame templates

Create and configure event frame templates to ensure consistent definitions for event frames in your asset structure.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, click **Event Frame Templates**.
3. To create a new event frame template, choose one of the following actions:
   - On the toolbar, click **New Template**.
   - Right-click **Event Frame Templates** and click **New Template**.
   - Right-click in the **Event Frame Templates** pane and click **New Template**.
4. In the **General** tab of the Event Frame Template Properties window, enter a unique name and a description for the event frame template in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the event frame template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing event frame template, which you select from the drop-down list. For more information on base templates, see Base and derived templates.
Severity	Select a severity setting for event frames based on the template. You can choose **None** (default), **Information**, **Warning**, **Minor**, **Major**, and **Critical**. **Note:** Event frames that have already been created with a specific severity setting are not updated automatically if the **Severity** setting in an event frame template is later changed.

Option	Description
Categories	Optional. Organize event frame templates by grouping them into element categories. To browse available categories or to create a new category, click .
Default Attribute	Optional. After you have created attributes for the template, you can select a default attribute from the drop-down list. For more information, see Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current local start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCSTARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute template, represented by the path. ◦ **%ELEMENT%** The name of the primary referenced element of the object's event frame. ◦ **%ELEMENTDESCRIPTION%** The description of the primary referenced element of the object's event frame. You can also use other naming pattern substitution parameters. For example, if you want an event frame generation analysis name included in the naming pattern, enter `%ANALYSIS%` as a substitution parameter. For a complete list of naming pattern substitution parameters, see Naming patterns. Each element derived from the template will have a unique, identifiable name. To ensure that new elements created from the template have an incremental number, enter * at the end of any pattern you enter here. Naming patterns are re-evaluated on an event frame when the **Start time**, **End time**, or primary referenced element is modified after it has been created. It will not be re-evaluated when loaded from another application. To force an element or event frame to re-evaluate its naming pattern, right-click the object and click **Reevaluate Naming Pattern**. Some substitution parameters may not be applied when a derived event frame is created. To ensure a derived event frame's name fully reflects the naming pattern, right-click the event frame and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters. If left blank, PI System Explorer uses the following rules: ◦ Event frames that are generated by event frame generation analyses use a default naming pattern of: `%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff%` ◦ Event frames that are created by other methods use the name of the event frame template and the current date: `%TEMPLATE%yyyyMMdd`

Option	Description
Allow Extensions	Select this check box to enable additional attributes to be defined for an event frame, beyond those defined in its template. For more information, see Extensions. **Note:** Categories, analyses and notifications are not affected by whether the **Allow Extensions** check box is enabled.
Can Be Acknowledged	Select this check box to enable users to acknowledge an event frame. For more information, see Acknowledgement of event frames. **Note:** This setting only affects event frames that are created after a template is created or modified.
Location	Click this link if you wish to specify location attribute traits for the event frame template. For more information, see Set location attribute traits.
Reason	Click this link if you wish to set up reason attribute traits for the event frame template so that users can select reason codes for downtime, excursions, and other events. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the event frame template beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.

Change event frame templates

You can change the template on which an existing event frame is based, or add a template to an event frame that is not currently based on a template.

Before you start

Exercise caution when you change an event frame template, since there can be unintended consequences. For example, attributes could be deleted if they were defined by the old template and are not present in the new template. Be sure that attributes with the same name in both the old and new template have the same data type.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, right-click the **Event Frame Searches** collection and click **New Search**.
3. In the Event Frame Search window, enter criteria to locate the event frame that you want to change.
4. Right-click the appropriate event frame in the browser and click **Convert** > **Change Template**.

5. In the Choose Event Frame Template window, select the desired template from the **Event Frame Template** list.
   a. Optional. To display only templates in a specific category, select a category from the **Templates of category** list.
6. Click **OK**.

Create event frames

Although you typically use event-frame-generation analyses or PI Event Frames Generator (PI EFGen) to create event frames, you can create an event frame manually. It is simply not recommended.

Procedure

1. In the navigator, select **Event Frames**.
2. To create a new event frame, choose one of the following actions:
   - On the toolbar, click **New Event Frame**.
   - In the Event Frames browser, right-click the **Event Frame Searches** collection and click **New Event Frame**.
   - In the viewer, right-click and click **New** > **New Event Frame**.
3. In the **Event Frame Template** section of the Choose Event Frame Template window, select an event frame template and click **OK**.

   The new event frame is displayed in the viewer with four tabs for configuring the event frame.
4. In the **General** tab, enter a unique name and a description for the event frame in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the event frame. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Template	Displays the template that you selected when you created the event frame. To review the template properties, click .
Severity	Select a severity setting for the event frame. You can choose **None** (default), **Information**, **Warning**, **Minor**, **Major**, and **Critical**.
Start time	Defaults to current date and time. Press F2 or click to open the Date and Time Picker window where you can select a different start time.
End time	Defaults to no date or time. Press F2 or click to open the Date and Time Picker window where you can select a different end time.
Categories	Optional. You can organize event frames by grouping them into element categories. To browse available categories, click .

Default Attribute	Optional. This field is read-only if the event frame is based on a template. After you have created attributes for the event frame in the **Attributes** tab, you can select a default attribute from the drop-down list. For more information, see Default attribute.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Annotations	Click this link if you wish to enter a comment and/or add a file attachment specific to the event frame. For more information, see Add annotations to event frames.
Location	Click this link if you wish to specify location attribute traits for the event frame. For more information, see Set location attribute traits.
Reason	Click this link if you wish to set up a reason attribute trait for the event frame. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the event frame beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Optional. You can choose from the following actions, as needed.

Option	Description
Capture values	To improve performance, you can save event-frame attribute values in PI AF rather than have data references calculated and executed. For more information, see Value capture for event frames and transfers.
Lock	After the underlying action for an event frame has completed, such as batch completion, you can click this link to prevent further changes to it. For more information, see Lock event frames or transfers. **Note:** Only Administrator users can unlock an event frame.
Acknowledge	If the event frame is based on a template where both the **Can Be Acknowledged** setting and the annotation access right are enabled, click this link to indicate that you have acknowledged the event frame. For more information, see Acknowledgement of event frames.

7. Edit the remaining tabs as needed.
   - To add attributes, see Create attributes on event frames.
   - To set a primary referenced element, see Primary referenced element.

Create attributes on event frames

Create attributes on event frames in the same way as on elements.

Procedure

1. In the Event Frames browser, find and select the parent event frame.
2. In the viewer, select the **Attributes** tab.
3. If no attributes exist, click the **New Attribute** link. Alternatively, click **New Attribute** on the toolbar.

   Note:
   If the event frame is based on a template, you cannot add an attribute unless the template allows extensions.

4. In the **Name** field, enter a unique name for the attribute.
5. In the **Description** field, enter a description for the attribute.
6. In the **Properties** field, select attribute properties as needed. For more information, see Attribute properties.
7. Optional. In the **Categories** field, assign the attribute to a category. To browse available categories, click .
8. In the **Default UOM** field, select the unit of measure for the attribute.
9. In the **Value Type** field, select the data type of the attribute.
10. To assign a value to the attribute, choose from the following actions.

To ...	Do this ...
Enter a static value	Type a value in the **Value** field.
Derive or calculate a value from a data reference	a. Select a data reference type from the **Data Reference** list. b. Click **Settings** to configure the data reference. For details about configuring a data reference, see Configuration of data references.

11. Optional. When the **Use DisplayDigits for Attribute and Attribute Template values** system option is selected, you can override the default DisplayDigits property value of -5 and specify the number of digits to display in the **Display Digits** field. Only numeric values in the **Value** field are affected. For attributes that are not based on a template, the precision of PI point data references is obtained from the PI point itself.
    - Enter a negative number up to -20 to indicate the total number of digits to display.
    - Enter a positive number up to 10 to indicate the number of digits after the decimal point to display, including trailing zeros.

    For more information, see Control the display of attribute and attribute template values and PI System Explorer customization options.
12. To save your work, press CTRL-S or click **Check In**.

Value capture for event frames and transfers

To display values for an event frame or transfer, PI AF executes data references to retrieve associated attribute values for the time period. You can, however, use **Capture Values** to save those values in a PI AF database table (`AFEventFrameAttributeValue`). Doing this can improve performance, since it is faster to display the saved values than to retrieve them.

You can also use **Capture Values** to ensure attribute values displayed for event frames are the same at any future point as when they were captured. Additionally, **Capture Values** provides a way to preserve values that might not be available later, such as streaming data that is not persisted long-term.

Value recapture

Automatic recapture occurs whenever the start time and end time of a captured event frame or transfer is modified.

Whenever you add new attributes to an event frame or transfer, you must recapture values to ensure that values are also captured for the new attributes.

Note that when you recapture values, data loss can occur if a data reference has changed since the initial value capture: For example, if values in a table accessed by a table lookup data reference have been modified.

Capture values

You capture values to save event frame or transfer attribute values to a table in the PI AF database. This can improve performance since PI AF does not execute any data references.

> **Note:**
> If you add new attributes to event frames or transfers with captured values, you should recapture those values to ensure that values are also captured for the new attributes.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, choose from the following actions.

To capture values for ...	Do this ...
All transfers or parent event frames in a collection	a. Right-click a search or recent collection. b. Click **Capture or Recapture Values**. Alternatively, follow these steps. a. Click a search or recent collection. b. In the viewer, right-click below the list of event frames or transfers. c. Click **Capture or Recapture Values**.

Child event frames in a collection	a. Click a search or recent collection. b. In the viewer, click ⊞ beside each event frame for which you want to capture values. The child event frames are displayed. c. In the viewer, use Windows selection keystrokes to select event frames and child event frames: ▪ To select contiguous objects, press SHIFT +<click>. ▪ To select non-contiguous objects, press CTRL+<click>. ▪ To select all displayed objects, press CTRL +A. d. Right-click a selected event frame. e. Click **Capture or Recapture Values**.
A single event frame	a. Expand a search or a recent event frame collection. b. Select the event frame. c. Choose from the following actions. ▪ In the **General** tab, click the **Capture Values** link. ▪ Right-click and click **Capture or Recapture Values**. **Note:** For subsequent value captures, the link changes to **Recapture Values**.
A single transfer	a. Expand a search or a recent transfer collection. b. Select the transfer. c. Right-click and click **Capture or Recapture Values**.

3. In the Capture or Recapture Values window, check the status and click **Close**.

Results

A Has Captured Values icon (📌) is displayed in the viewer beside each event frame or transfer that has a captured value.

Lock event frames or transfers

After the underlying action for an event frame or transfer has completed (for example, after a batch has finished), the event frame or transfer is still open and can continue to receive data. You can lock an event frame or transfer, preventing further changes to it.

Unlocking an event frame or transfer requires Administrator permission. Only users with Administrator permissions on an object have the ability to unlock it.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, open the **Event Frame Searches** (or **Transfer Searches**) collection that contains the object you want to lock.
3. Right-click the object and click **Lock**.
4. To verify that the object is locked, select the event frame or transfer search collection that contains the object.

   The object is displayed in the viewer with next to it in the Lock column.

Acknowledgement of event frames

In PI AF 2016, you can require that an event frame be acknowledged. The acknowledgement feature is used by notifications and PI AF client applications, such as PI Vision. The event-frame acknowledgement feature replaces the acknowledge notification functionality in the legacy version of Notifications.

- For more information on setting up event-frame acknowledgement in notifications, see Configuration of notification rules for analyses or event frames.
- For more information on viewing and acknowledging event frames using PI Vision, see the PI Vision topic "View event details and annotate events" in Live Library (https://livelibrary.osisoft.com).

Security for event-frame acknowledgement

A user needs to have a PI AF identity for which the Annotate permission (an) is enabled. For more information on access rights, see PI AF access rights.

Setup of event-frame acknowledgement

You set up an acknowledgement by selecting the **Can Be Acknowledged** check box as you configure an event frame template. Any event frame that is based on that template displays an **Acknowledge** link on the **General** tab. In event frames based on event frame templates where the **Can Be Acknowledged** check box is not selected, the **Acknowledge** link is inactive. You also cannot acknowledge event frames that are not based on an event frame template.

> **Note:**
> The **Can Be Acknowledged** setting only affects event frames that are created after a template is created or modified.

Search for event-frame acknowledgements

You can search for event frames that can be acknowledged, as well as those that have been acknowledged. You review acknowledgement status in the Viewer for the Event Frame Searches collection, where an Acknowledged column is displayed ( ). Possible values are:

	Is not acknowledged
	Is acknowledged
blank	Not acknowledgeable

To group event frames by their acknowledged status, click the column icon.

Acknowledge event frames

You acknowledge event frames in the Event Frames browser or in the **General** tab of a selected event frame in the Event Frames viewer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click an Event Frame Search collection. The contents of the collection are displayed in the viewer.
3. Optional. To group event frames by their acknowledged status, click the ⚠ column icon.
4. Choose from the following actions.

To ...	Do this ...
Acknowledge an event frame	a. Choose one of the following actions: ▪ Right-click an event frame in the viewer and click **Acknowledge** on the context menu. ▪ Double-click an event frame in the viewer and, on the **General** tab, click the **Acknowledge** link. b. Click **Yes** in response to the **Are you sure you want to acknowledge *Event_Frame_Name*?** prompt. c. The following changes occur: ▪ In the Event Frame Search collection that is displayed in the viewer, the Acknowledged (⚠) column changes to OK status. ▪ On the **General** tab, the **Acknowledged** field is displayed with a time and date stamp. The user ID of the user who made the acknowledgement is displayed in the **By** field.
Review an acknowledgement	a. In the Event Frame Search collection that is displayed in the viewer, double-click an event frame that has an OK status. b. On the **General** tab, review the time and date stamp in the **Acknowledged** field, as well as the user ID of the user who made the acknowledgement.

Transfers

Transfers are a type of event frame. They mark movement of material in discrete quantities. They have a start time and an end time. Transfers are unique in a model because they are temporal and appear in a model only when a transfer of material has taken place. For example,

use transfers to track material movements in and out of the facility, track raw materials used in the process and finished product being stored, and track tank-to-tank material transfers.

Topics in this section

- Create transfer templates
- Create transfers
- Create transfer ports

Create transfer templates

Create and configure transfer templates to ensure consistent definitions for transfers in your asset structure.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, click **Transfer Templates**.
3. To create a new transfer template, choose one of the following actions:
   - On the toolbar, click **New Template**.
   - In the **Transfer Templates** pane, click **New Template**.
   - Right-click **Transfer Templates** and click **New Template**.
4. In the **General** tab of the Transfer Template Properties window, enter a unique name and a description for the transfer template in the **Name** and **Description** fields.
5. Adjust settings on tabs to configure the transfer template. PI System Explorer provides defaults for all required settings, but you can configure settings yourself. The settings include:

Option	Description
Base Template	You can base the template on an existing transfer template, which you select from the list. For more information on base templates, see Base and derived templates.
Categories	Optional. Organize transfer templates by grouping them into element categories. To browse available categories or to create a new category, click .
Default Attribute	Optional. After you have created attributes for the template, you can select a default attribute from the list. For more information, see Default attribute.

Option	Description
Naming Pattern	Optional. You can enter a text string, or click ▸ to choose from the following substitution parameters to define a naming pattern. ◦ **%TIME:yyyy-MM-dd HH:mm:ss.fff%** The current local time of the object using the specified formatting. If no formatting is specified, the default formatting is used. The formatting uses standard format strings supported by the `DateTime.ToString` method, as described in Format strings for time substitution parameters. ◦ **%UTCTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current local start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%UTCSTARTTIME:yyyy-MM-dd HH:mm:ss.fff%** The current Coordinated Universal Time (UTC) start time of the object using the specified formatting. If no formatting is specified, the default formatting is used. ◦ **%TEMPLATE%** The name of the object's template. ◦ **%@Attribute%** The value of the object's attribute template, represented by the path. ◦ **%SOURCE%** Name of the source element for the transfer. ◦ **%DESTINATION%** Name of the destination element for the transfer. You can also use other naming pattern substitution parameters. For a complete list of naming pattern substitution parameters, see Naming patterns. Each transfer derived from the template will have a unique, identifiable name. To ensure that new transfers created from the template have an incremental number, enter * at the end of any pattern you enter here. Naming patterns are re-evaluated on a transfer when the **Start time** is modified after it has been created. It will not be re-evaluated when loaded from another application. To force a transfer to re-evaluate its naming pattern, right-click the object and click **Reevaluate Naming Pattern**. Some substitution parameters may not be applied when a derived transfer is created. To ensure a derived transfer's name fully reflects the naming pattern, right-click the transfer and click **Reevaluate Naming Pattern**. **Note:** The name generated by the naming pattern must be less than the maximum name length of 260 characters. If left blank, PI System Explorer uses the name of the transfer template and the current date: `%TEMPLATE%yyyyMMdd`

Option	Description
Allow Extensions	Select this check box to enable additional attributes to be defined for a transfer, beyond those defined in its template. For more information, see Extensions. **Note:** Categories are not affected by whether the **Allow Extensions** check box is enabled.
Extended Properties	This link is an advanced feature. For more information, see Storage of application-specific information.
Reason	Click this link if you wish to set up a reason attribute trait for the transfer template. For more information, see Set reason attribute traits.
Security	Click this link if you wish to set up custom access permissions to the transfer template beyond those already established at the server and database level. For more information, see Configure security for objects.

6. Use the **Attribute Templates** tab to create an attribute template for each property or data item for the template. See Create attribute templates.
7. Optional. Use the **Ports** tab to define additional ports that define end-points for connections between transfer elements, or to modify the default **In** and **Out** ports. See Create transfer ports.

Create transfers

You create a transfer in the Event Frames browser.

Before you start

Default **In** and **Out** transfer ports with Boundary and Node connection types are automatically created for every transfer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click the **Transfer Searches** (or **Recent Transfers**) collection.
3. To create a new transfer, choose one of the following actions:
   - On the toolbar, click the **New Transfer** button.
   - In the **Event Frames** browser, right-click the **Transfer Searches** collection and click **New Transfer**.
   - In the viewer, right-click and click **New Transfer**.
4. In the Choose Transfer Template window, choose an existing template for the transfer. Although not recommended, you can also choose **<None>**.
   a. Optional. To assign the transfer to a category, select one from the **Templates of category** list.
5. Click **OK**.

   The new transfer is displayed in the viewer with three tabs for configuring the transfer. In the **General** tab, a default name for the new transfer is displayed based on the naming pattern, as well as a default start time.

6. Optional. Replace the default name with a unique name and enter a description for the transfer in the **Name** and **Description** fields.
7. Optional. You can organize objects by grouping them into categories. To browse the available categories, click . See Categorization of objects for more information.
8. In the **Start time** and **End time** fields, enter the start and end times for when the transfer takes place. To browse for a date, click .
9. In the **Source** and **Port** fields, select the element and port providing the material of the transfer.
10. In the **Destination** and **Port** fields, select the element and port receiving the transfer.
11. If the transfer is based on a template, a read-only value is displayed in the **Default Attribute** field. However, if you are creating a transfer manually, you can select a default attribute after you have added attributes in the **Attributes** tab.
12. Optional. To add an annotation to the transfer, click **Annotations**. For more information, see Add annotations to transfers.

After you finish

If the **Allow Extensions** check box is selected in the transfer template or you are creating a transfer manually, follow these additional steps.

- Use the **Attributes** tab to define an attribute for each property or data item. See Create attribute templates.
- Use the **Ports** tab to define additional ports that define end-points for connections between transfer elements, or to modify the default **In** and **Out** ports. For more information, see Create transfer ports.

Create transfer ports

Use the **Ports** tab to review settings for default **In** and **Out** transfer ports.

Before you start

You can only modify or add ports for a transfer derived from a template if **Allow Extensions** is enabled on the transfer template.

Procedure

1. To create a new transfer port or review existing ports, choose one of the following actions:
   - In the Event Frames browser, expand a **Transfer Searches** collection. Select a transfer, or right-click a transfer and click **Properties**.
   - In the Library browser, expand **Transfer Templates**. Select a transfer template, or right-click a transfer template and click **Properties**.
2. Click the **Ports** tab.
3. Choose from the following actions.

To ...	Do this ...
Create a new port	a. Choose one of the following actions. ▪ On the toolbar, click **New Port**. ▪ Right-click in the viewer and click **New Port**. b. Choose one of the following actions. Refer to Transfer Ports tab properties for descriptions of required values. ▪ Edit the values in the **Ports** columns directly. ▪ Right-click the port and click **Properties**. Enter values and click **OK** to save your changes.
Modify an existing port	Choose one of the following actions. Refer to Transfer Ports tab properties for descriptions of required values. ◦ Edit the values in the **Ports** columns directly. ◦ Right-click the port and click **Properties**. Edit values and click **OK** to save your changes.
Specify a default port	a. Right-click a port that displays the icon. b. Select **Set as Default Port**.
Modify a default port	a. Right-click a port that displays the icon. b. Select **Properties**. c. In the Port Properties window, clear the **Default Port** check box. d. Click **OK**.

4. Check in your changes.

Transfer Ports tab properties

The **Ports** tab for a transfer contains the following columns.

Field	Description
Name	The name of the port.
Description	Optional. The description of the port.
Port Type	Indicates the type of port: **Input**, **Output**, or **Undirected** (for meters, for example).
Allowed Categories	The categories to which the port belongs. To assign categories, click and select categories in the Categorize window.
Maximum Connections	The maximum number of connections that can be made to the port. The default number of connections for a transfer is 1. A value of zero indicates an unlimited number of connections.
Connection Type	The type of connections to which the port can connect. The default connection types for a transfer are **Boundary** and **Node**. For more information on connection type values, see Element types in models.
Allowed Template	Defaults to all element templates, but you can select a specific template from the list.

Annotations in PI AF

Beginning with PI AF 2016, the annotation feature is used by client applications such as PI Vision, but can easily be used by administrators to make notes on the following objects:

- Case
- Element
- Event frame
- Model
- Transfer

File attachments

Users can attach a single file to an annotation. By default, the following file types are allowed:

File type	Allowed extension
MS Office	csv, docx, pdf, xlsx
Text	rtf, txt
Image	gif, jpeg, jpg, png, svg, tiff
ProcessBook	pdi

The maximum file size defaults to 10MB.

Administrators can use the PI AF Diagnostics utility (`afdiag` located in the `\PIPC\AF` folder) to specify the file types that can be attached to annotations (with the **`FileExtensions`** parameter), as well as the maximum file size (with the **`FileMaxLength`** parameter).

Security for annotations

A user needs to have a PI AF identity for which the Annotate permission (**`an`**) is enabled. For more information on access rights, see PI AF access rights.

Topics in this section

- Element annotations
- Event frame annotations
- Transfer annotations

Element annotations

Users can annotate an element, as well as upload a file to link to an annotation.

Setup of element annotations

You set up an annotation by opening an Annotations window for a specific element that you have selected in the Elements browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Review of element annotations

You review annotation status for an element in the Elements viewer, where an Is Annotated column is displayed ( ).

To view the content of an annotation, move the mouse pointer over the icon beside an element to see the annotation text. If a file is attached, the name of the file is also displayed.

Add annotations to elements

You can add one or more annotations to an element.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, choose one of the following actions.
   - Click the **Elements** collection.
   - Navigate to the specific element you want to annotate.
3. To add a new annotation, choose one of the following actions.
   - In the Elements browser, right-click an element and click **Annotate**.
   - If an element is selected in the Elements browser, click the **Annotations** link on the **General** tab in the viewer.
   - If the **Elements** collection is selected in the browser, right-click an element in the viewer and click **Annotate**.

     Note:
     To add another annotation to a previously annotated element in the viewer, double-click beside the element.
4. In the Annotations window, click **New Annotation**.
5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.
6. To attach a file to the annotation, click **Add Attachment**.
   a. In the Open window, navigate to the directory where the file you wish to upload is located.
   b. Select the file and click **Open**.
   The filename is displayed in the **Attachment** field.
7. Click **Close**.

Results

When elements are listed in the Elements viewer, an Is Annotated column is displayed beside the element. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Event frame annotations

Users can annotate an event frame, as well as upload a file to link to an annotation.

Setup of event frame annotations

You set up an annotation by opening an Annotations window for a specific event frame that you have selected in the Event Frames browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Search for event frame annotations

You can search for event frames that have been annotated. You review annotation status in the viewer for an Event Frame Searches collection, where an Is Annotated column is displayed ( ).

To group event frames by their annotation status, you click the column icon.

Add annotations to event frames

You can add one or more annotations to an event frame.

Before you start

To annotate a locked event frame, a user with Administrator permission must first unlock it.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click an **Event Frame Searches** collection. The contents of the collection are displayed in the viewer. Event frames with existing annotations display an Is Annotated icon ( ) beside them.
3. To add a new annotation, choose one of the following actions.
   - In the viewer, right-click an event frame and click **Annotate**.
   - In the viewer, double-click an event frame and click the **Annotations** link on the **General** tab.
   - In the Event Frames browser, right-click an event frame in an event frame search collection and click **Annotate**.
   - To add another annotation to a previously annotated event frame, double-click beside the event frame.
4. In the Annotations window, click **New Annotation**.
5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.

6. To attach a file to the annotation, click **Add Attachment**.

a. In the Open window, navigate to the directory where the file you wish to upload is located.

b. Select the file and click **Open**.
The filename is displayed in the **Attachment** field.

7. Click **Close**.

Results

In the Is Annotated column, is displayed beside the event frame. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Transfer annotations

Users can annotate a transfer, as well as upload a file to link to an annotation.

Setup of transfer annotations

You set up an annotation by opening an Annotations window for a specific transfer that you have selected in the Event Frames browser or viewer. In the Annotations window, you can add, edit, and delete annotations, as well as upload, change, and delete an attachment for an annotation.

Review transfer annotations

You can review annotation status in the viewer for a Transfer Searches collection, where an Is Annotated column is displayed ( ).

To group transfers by their annotation status, you click the column icon.

Add annotations to transfers

You can add one or more annotations to a transfer.

Procedure

1. In the navigator, click **Event Frames**.
2. In the Event Frames browser, click a **Transfer Searches** collection. The contents of the collection are displayed in the viewer. Transfers with existing annotations display an Is Annotated icon ( ) beside them.
3. To add a new annotation, choose one of the following actions.
   - In the viewer, right-click a transfer in the search result list and click **Annotate**.
   - In the viewer, double-click a transfer and click the **Annotations** link on the **General** tab.
   - In the Event Frames browser, right-click a transfer in a **Transfer Searches** collection and click **Annotate**.
   - To add another annotation to a previously annotated transfer, double-click beside the transfer.
4. In the Annotations window, click **New Annotation**.

5. Enter an annotation in the **Comment** field.

   Note:
   If you require more space as you type, press F2 and enter an annotation in the Text Visualizer window. Click **OK** to close the window.

6. To attach a file to the annotation, click **Add Attachment**.

   a. In the Open window, navigate to the directory where the file you wish to upload is located.

   b. Select the file and click **Open**.
   The filename is displayed in the **Attachment** field.

7. Click **Close**.

Results

In the Is Annotated column, is displayed beside the transfer. Move the mouse pointer over the icon to see the annotation text. If a file is attached, the name of the file is also displayed.

Asset analytics

Asset analytics is the feature in PI Asset Framework (PI AF) that you use to create and manage analyses. An analysis reads values of PI AF attributes, performs calculations, and writes results to other attributes. Within PI System Explorer, you can access asset analytics when you work with elements and element templates.

Topics in this section

- Introduction to analyses
- Expression analyses
- Rollup analyses
- Event frame generation analyses
- SQC analyses
- Sample analyses
- Backfilling and recalculation of analyses
- Management of analyses in a PI AF database
- Expression functions reference
- Steam functions for analysis expressions
- Asset analytics native integration of external systems
- PI Analysis Service management

Introduction to analyses

An analysis performs calculations on a specified schedule. An analysis takes existing PI AF attribute values as inputs and produces new outputs, either new calculated values or new event frames. Every analysis has an associated element or element template.

The following types of analyses are available:

- **Expression**

  Calculates one or more output values from specified functions, operators, and input values.

- **Rollup**

  Calculates standard statistical functions for a group of selected attributes. This group is selected from attributes on an element or from the set of all attributes on its sub-elements.

- **Event frame generation**

  Starts or ends event frames based on specified conditions.

- **SQC**

  Uses Statistical Quality Control (SQC) methods to monitor that attribute values lie within predetermined boundaries.

All analyses can be created from analysis templates with either `Enabled` or `Disabled` status; by default, analyses will be enabled when created. To create an analysis with `Disabled` status, make sure to clear the **Enable analyses when created from template** check box in the analysis template.

For event frame generation analyses and SQC analyses with event frame outputs, you can create a notification rule template by clicking **Create a new notification rule template for *Analysis Template Name***.

For expression analyses and event frame generation analyses, you specify inputs in expressions.

For event frame generation analyses (running PI Analysis Service 2017 R2 or later versions), expression and rollup analyses, you can map analysis outputs to PI AF attributes. If you map outputs to attributes configured as PI point data references, the analysis saves the history of the output to a PI point. OSIsoft recommends that you save analysis outputs to PI points. You can use the PI point in visualization tools to view the trend of the analysis. Saving outputs to PI points also results in better PI AF performance than saving outputs to attributes without data references.

Note:
PI Analysis Service does not use exception reporting while writing to output configured as PI point data references. Although exception settings for the output PI point are not used, compression settings for the output PI point will be used to determine the outputs that are written to PI Data Archive.

Topics in this section

- Configure and manage analyses on an element
- Analysis templates
- Updates of analyses and analysis templates
- Excluded attributes in analyses
- Analysis scheduling
- Expressions for analyses
- Future data and analyses

Configure and manage analyses on an element

A newly-created analysis is listed in the table in **Analyses** tab. You can see its current status, whether or not it is associated with a template, its analysis type, name, and backfilling/recalculation status. Hover over items or icons in the list to see descriptive tooltips. To manage multiple analyses in the **Management** pane, see Management of analyses.

To view the expression and scheduling for an analysis, select it from the **Analyses** viewer.

You can perform these operations from the **Analyses** viewer:

- **Create a new analysis**

  You can click (New Analysis) at the top of the viewer, or right-click anywhere in the viewer and click **New**.

- **Enable or disable a selected analysis**

  You can select an analysis and click (Enable Analysis) or (Disable Analysis) at the top of the viewer.

- **Choose columns to display**

  You can right-click any column heading and select headings to display.

- **Preview results**

  You can preview results for an analysis to see how it operates over a specified time range by right-clicking an analysis and selecting **Preview Results**. You can export the results table to a spreadsheet or you can copy selected rows from the results table into other applications.

  For information on viewing trend charts for output times that are specified relative to trigger times, see Trend charts in Preview Results.

- **Backfill or recalculate data for an analysis**

  You can execute an analysis over an earlier time period to backfill or recalculate data, as described in Backfill or recalculate data for an analysis.

  An expression, rollup, or SQC analysis can backfill or recalculate data if the analyses outputs are mapped to PI point attributes. You can backfill the missing data in a time range and retain existing data, or you can recalculate, which deletes and recalculates data in the time range.

  An event frame generation analysis can also recalculate event frames over a specified time period. The recalculation process, however, automatically deletes all existing event frames for that time period, as well as annotations on affected event frames.

  In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. Note that recalculation is only available for PI Analysis Service 2016 R2 or later. In addition, PI Data Archive that stores PI points must be running version 2016 R2 or later.

- **Enable automatic recalculation for an analysis**

  From PI AF 2017 R2, you can automatically recalculate analyses. Although automatic recalculation is globally enabled by default, you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation.

  An analysis listed with means automatic recalculation is enabled. See Configure automatic recalculation of analyses and analyses templates for more information.

- **Go to template**

  An analysis listed with alongside is based on an analysis template. You can open the template directly from the analysis by right-clicking the analysis and selecting **Go to Template**.

  > **Note:**
  > You can go to the analysis directly from the attribute by right-clicking the attribute and clicking **Go to Analysis**. If there is no analysis configured on the attribute, you will see **New Analysis** instead.

- **Audit Trail Events**

  You can open the audit log directly from the analysis by right-clicking the analysis and selecting **Audit Trail Events**. For more information on audit trail, see Track PI AF changes with Audit Trail.

- **Reset to template**

  Only scheduling changes can be made to an analysis that is based on an analysis template. To revert an analysis' scheduling to template scheduling, right-click the analysis and click **Reset to Template**.

- **Convert to template**

  If the element for an analysis is derived from an element template, you can convert the analysis to an analysis template on that element template by right-clicking the analysis and selecting **Convert to Template**. This feature enables you to develop and debug an analysis against a specific element before generalizing it as a template.

  > **Note:**
  > If any of the analysis outputs write to a specific PI point, you will be prompted to choose how to specify the PI point in the analysis template to ensure the outputs of derived analyses write to unique PI points.

To create specific types of analyses, see:

- Create an expression analysis
- Create a rollup analysis
- Create an event frame generation analysis
- Create an SQC analysis

Analysis templates

To apply an analysis to an element, you specify the element directly as you create the analysis. However, to apply an analysis to a group of elements (created from the same element template), it is much more efficient to use an *analysis template* than to apply the analysis individually to each element.

An analysis template defines the form of an analysis. Analyses derived from it are all similar but have specific input and output attributes. Analysis templates let you take advantage of your PI AF hierarchy. Every element derived from an element template automatically acquires analyses from its analysis templates.

You create an analysis template on an element template in much the same way you create an analysis on an element. When you add or change an analysis template on an element template, those changes propagate to every element derived from the element template. Deleting an analysis template deletes all analyses derived from it, except for analyses tied to notifications.

With PI Server 2015 R2 and later versions, while creating an analysis template on an element template that is derived from a base element template, you can choose input attributes from attribute templates that are defined on derived as well as base element templates. Similarly, you can map analysis outputs to derived as well as base template attributes.

> **Note:**
> An analysis derived from an analysis template cannot be removed from an element directly.

To create specific types of analysis templates, see:

- Create an expression analysis template
- Create a rollup analysis template
- Create an event frame generation analysis template
- Create an SQC analysis template

Updates of analyses and analysis templates

As soon as a new analysis is checked in, it is automatically enabled and available to run. Any change that is checked in is immediately picked up by all the analyses affected by it, even if they are running. Changes that can affect an analysis include:

- Edits made directly to the analysis
- Changes to an element that impact its analyses, including adding or deleting an analysis
- Changes to an element template, which propagate to all elements that come from the template, including adding or deleting an analysis template
- Changes to an analysis template, which propagate to all analyses that come from the template
- Changes to an event frame template used by an analysis; new event frames are based on the updated template

Excluded attributes in analyses

In PI Asset Framework 2015, excluded attributes were introduced. For more information on excluded attributes, see Excluded attribute property. Beginning with PI AF 2018, behavior and handling of excluded attributes have changed in expression, event frame generation and SQC analyses.

Prior to 2018, analyses configured with excluded attributes would return **Calc Failed** when evaluated and go into error. Beginning with PI AF 2018, such analyses will be suspended and marked by an icon (⏸) in the Analyses Viewer. If you hover over the icon, you will see tooltip with more information.

If you do not want the analyses with excluded attributes to go into a suspended state, you can validate the excluded attribute with an expression function **BadVal**. Assuming 'att1' is an excluded attribute, below is an example of how **BadVal** could be used to prevent the analysis from being suspended. In this case, the analysis will continue to run:

```
If BadVal('att1') Then ('att2') Else ('att1' + 'att2)
```

> **Note:**
> Any change in the configuration of the excluded attribute in the analysis is automatically picked up by PI Analysis Service. For example, if you decide to change the property of 'att1' to be no longer excluded and check it in, the service will detect it and run the analysis with 'att1.'

When you select **Event-Triggered** as a scheduling option, you are required to select a triggering input. Note that if the excluded attribute is the only available triggering input, the analysis will be suspended whether or not you validate it with **`BadVal`**. It is because analyses do not trigger on excluded attributes. Make sure that you have other inputs to trigger on in your analyses. For more information, see Analysis scheduling.

Analysis scheduling

Each analysis requires you to specify scheduling. Scheduling indicates when to evaluate an analysis automatically. There are two types of scheduling:

- **Periodic**

  With periodic scheduling, evaluation occurs based on clock time. You can specify a specific time for evaluation each day or the time between evaluations. With PI Server 2015 R2 and later versions, the maximum frequency you can set for analyses is 120 times per second.

  Note:

  - If you have analyses that are previously configured to run more than 120 times per second, you must reduce the period to less than the allowed maximum, else they will generate errors while running with PI Server 2015 R2. Similarly, if you are setting periodic scheduling outside of the PI System Explorer user interface, make sure that the period is set to less than the allowed maximum.
  - If the period is not divisible by the 24, 7 minutes for example, the evaluation time will reset at the beginning of each day. Let's say you have scheduled the analysis to run every 7 minutes and the last evaluation was at 11:56pm. The next evaluation will be at midnight (4 minutes later), not at 12:03am the next day.

- **Event-Triggered**

  With event-triggered scheduling, evaluation occurs based on events. You can specify one or more input attributes that trigger an evaluation whenever the attribute value changes.

  Note:
  When auto-backfilling is enabled, triggering events with time stamps before analysis service start-time are ignored for the purpose of real-time evaluation. For more discussion on real-time evaluation and auto-backfilling, including the setting of the **`AutoBackfillingEnabled`** configuration parameter, see Analysis service configuration.

By default, analyses use event-triggered scheduling, triggered on changes to any input.

Many factors affect the speed that an analysis runs and writes output values:

- System architecture, including inputs and outputs
- Network configuration
- Load and performance of different computers

Monitor and test your analyses, especially those scheduled at high frequencies, to ensure the system resources support the configuration.

Output time stamps for analyses

For any scheduled analysis, the default time stamp for output values is the *trigger time.* For periodic scheduling, the trigger time is the scheduled evaluation time, and for event-triggered scheduling, the trigger time is the time that a specified attribute changes values.

You can specify the time stamp of analysis output values. By specifying an output time stamp, you can:

- Produce results that are time-stamped at a specified offset from the trigger time.
- Produce results that are time-stamped at a particular time each day, regardless of when the analysis is actually evaluated. For example, you can ensure a 6:00 p.m. time stamp for the output of an analysis that calculates results for a daily production shift ending at 6:00 p.m., even if the analysis is not actually evaluated until 10:00 p.m.
- Repeatedly evaluate an analysis to calculate a forecast value for a particular date and time, as input conditions change. The results of each evaluation have same time stamp: the analysis output values reflect results from the most recent evaluation.

Specify time stamp for analysis outputs

Use the Advanced options window in an analysis to specify the time stamp of analysis output values.

Procedure

1. In the Elements browser, select an element, and in the viewer, click the **Analyses** tab.
2. From the list of analyses, select an analysis, and then click **Advanced** in the scheduling area near the bottom of the window.
3. In the Advanced options window, specify the output time stamp:
   - **Trigger Time**

     Default value. The clock time that a schedule specifies or that an input value changes.
   - **Execution Time**

     The clock time at which the analysis calculates the value. This differs from the trigger time when the system has a lag time or when there is a configured calculation wait time (see **`CalculationWaitTimeInSeconds`** in Analysis service configuration).
   - **Relative to Trigger Time**

     A time specified by a PI time expression. Enter a valid time expression, such as a time relative to the trigger time or a fixed time. In the expression, the current time and implied reference time is the trigger time (that is, the clock time that the schedule specifies or that an input value changes). If you enter a fixed time, all output values from the analysis will have the entered time stamp. When not specified, default unit of measure for relative time is hours.

     For information on viewing trend charts for output times that are specified relative to trigger times, see Trend charts in Preview Results.
   - **Variable**

Use the value of a variable as an output timestamp. **Variable** must be a timestamp of value type `datetime`. If this value is calculated from an expression, PI Analysis Service uses the calculated value for each trigger events as the output timestamp. This option is only available for expression analyses.

You can preview your results by right-clicking the analysis and selecting **Preview Results**. For more information, see Trend charts in Preview Results.

Trend charts in Preview Results

When you right click an expression or rollup analysis and select Preview Results, you see the trend chart with input and output attribute values at different trigger times.

If you select **`Relative to Trigger Time`** and specify '*-8h' as output time stamp for example, you see two trend charts displayed, one for input and the other for output. The input chart shows the same time range as the configured start and end time; however, the output chart shows the time that is calculated using the output time stamp override to the trigger time. Each trend chart shows a maximum of four input and four output attributes; currently, the input or output attributes that are displayed on the chart are automatically selected by PI Analysis Service.

In the table, the column `Output Time` indicates the output time calculated with the time stamp override.

Note:
The Preview Results charts may depict data that differs from the actual results of the PI Analysis Service. For example, if the same timestamp data is overwritten several times, the Preview Results table and charts show each written value of the data point, but the PI Analysis Service only uses the last written value.

Expressions for analyses

For expression analyses and event frame generation analyses, you specify inputs and calculations in expressions. In PI System Explorer, you enter each expression in a row. Each row has the following columns:

- **Name**

  A variable name for the value computed by the row. This variable is internal to the analysis. You can use this variable as an input to expressions specified by subsequent rows in the analysis.

- **Expression**

  The specification of a calculation performed. The expression can include PI AF attributes, variables from earlier rows in the analysis, and functions.

- **Value at Evaluation**

  The current computed value of the expression. Click **Evaluate** to compute again.

- **Value at Last Trigger**

  The value when the last trigger was computed.

- **Output Attribute**

  For expression analyses, the attribute that stores the computed value. If the analysis contains multiple expressions, you can map the value from any expression to an output attribute. However, at least one expression must be mapped to an output attribute.

For example, an expression analysis might contain one expression, specified in a single row.

Name	Expression	Value at Evaluation	Output Attribute
Variable1	PrevVal('Att1') - 100	505	Analysis1_Output

You can also specify the calculation of an analysis with multiple expressions across several rows. For example, an analysis might contain three expressions: the first two calculate variables V1 and V2, which are used as inputs to the last expression.

Name	Expression
V1	'Att1'/2
V2	'Att2' - 'Att3'
Result	V1 + 3*V2

Analyses with multiple expressions evaluate the expressions in order: the top row first, the one below it next, and so on. Because expressions in lower rows may depend on results from higher rows, you can reorder the rows to evaluate them in the proper order.

You can click **Add a new variable** to add a row and the ⊗ icon to delete it.

Tip:

- You can add a line break within the expression by clicking shift+enter.
- You can make your expressions easier to read by adding comments next to or within an expression. Use double slash (//) to introduce a comment or place the comment between slash asterisk (/*) and asterisk slash (*/).

```
// Add comments here
IF x THEN y ELSE /*Do this if x is false*/ z
```

Name	Expression
Variable1	TimeEq('sinusoid', '*-24h' /* Indicates start time*/, '*' /*Indicates end time*/, 80) //Find the total time within the past 24 hours when the value of 'sinusoid' was 80)

Topics in this section

- Expression specification
- Expression simplification with refactoring

Expression specification

To create an expression, click in the **Expression** column of a row and then specify the input attributes, variables, and functions that define a calculation. You can do the following:

- Select and insert attributes and functions into the expression from the **Functions** and **Attributes** and panels to the right of the expression.
- Enter the text directly into the expression. As you type, functions, attributes, and variables that match what you entered appear in a selection list at the cursor: you can select an item you want to insert.

  Attributes and time expressions are enclosed in single quotes.

  Strings are enclosed in double quotes.

Expression simplification with refactoring

In real-world applications, the expression in an analysis can be extremely complex; lengthy attribute names can add to the complexity. To simplify a complicated expression, refactor it into a group of smaller expressions assigned to variables.

Refactored analyses are easier to understand. Refactored analyses are also easier to test and debug: you can evaluate the smaller expressions one at a time and isolate any errors.

To refactor an analysis, define components of the analysis as separate expressions and reference by variable name. For example, you might define a separate expression that defines a long attribute name as a separate variable or that defines a particular calculation as a separate variable. To create these separate expressions, simply add new expression rows to the analysis and enter text in the **Name** and **Expression** columns.

To reduce typing, you can select a term in an expression, right-click the selection, and then click **Assign to variable**. In the row above the expression, PI System Explorer inserts a new row that maps the selected term to a new variable name. In the original expression, PI System Explorer replaces the term with the new variable name.

You can edit the name of any row. However, if you change the name of an expression row already used as a variable in other expressions, you break the connection between the expressions.

Example

Suppose you have a complex expression defined in a single row:

Name	Expression
`Variable1`	`2*'LongAttributeName' + Avg('Att2', 'Att3', 'Att4')`

You can refactor the expression into the following three rows that contain simpler expressions. Note that third row uses the names of the earlier rows as variables.

Name	Expression
`Variable1`	`'LongAttributeName'`
`Variable2`	`Avg('Att2', 'Att3', 'Att4')`
`Variable3`	`2*Variable1 + Variable2`

Future data and analyses

You can use future data as input to an analysis. You can also use an analysis to produce future data by specifying a future time stamp for the output from an analysis.

Expressions can use values from future events as input to an analysis. However, if an analysis uses event-triggered scheduling, inputs with future data can generate analysis evaluations. If a trigger attribute has future events, an evaluation occurs as each event becomes current, when the clock time coincides with the time of that event.

The following sections show two uses of future data in analyses. The first example shows how to use future data as an input to an analysis. The second example shows how to use an analysis to produce future data.

Example: Input future data to evaluate forecast values

Suppose you need to evaluate forecast values for the outside temperature provided by a third-party service. To do that, create an analysis that finds the difference between forecast and actual values.

The analysis needs two input attributes:

- **ActualTemp**

  Stores near real-time temperature readings from an outside thermometer.

- **ForecastTemp**

  Stores future data, in this case forecast values for the outside temperature each day at 6 a.m., 2 p.m., and 10 p.m.

The analysis calculates the difference between the `ActualTemp` and `ForecastTemp` values and writes results to an output attribute named `DeltaTemp`.

Choose event-triggered scheduling for the analysis with the `ForecastTemp` attribute as the only triggering input attribute. With event-triggered scheduling, an evaluation occurs when the analysis detects a new event for a trigger attribute. Because this trigger attribute stores future data, the analysis detects a new event three times a day, when the clock time reaches 6 a.m., 2 p.m., and 10 p.m.; this triggers an evaluation and a new result for the `DeltaTemp` attribute.

Example: Output future data to calculate weekly emissions goals

Suppose you need to set weekly goals for emissions at a cement plant that operates under a voluntary annual CO_2 cap. The weekly goal will likely change every week in response to the actual emission levels of the past week and to the amount remaining under the voluntary annual cap.

At the end of every week, you know actual emissions since the beginning of the year and can calculate a new weekly emissions goal. You want the time stamp for the new goal to be exactly one week from the calculation time, which coincides with the calculation time of the actual emissions for the next week.

In PI AF, create attributes for the parameters the calculations require, such as the annual cap.

Next, create a new expression analysis that calculates an output attribute named `WeeklyGoal`. The analysis completes the following steps:

1. Find the actual emissions since the beginning of the year.
2. Find the difference between the year-to-date total and the annual cap.
3. Divide the difference by the number of days remaining under the annual cap and multiply by seven to calculate the `WeeklyGoal` attribute.
4. Save the `WeeklyGoal` attribute as a future PI point, which preserves history and allows for trending and other uses.

When creating the analysis, choose periodic scheduling for every day at 12:00 a.m. The syntax of the analysis ensures the calculations only run on Sunday.

Finally, specify the time stamp for the analysis outputs. Enter the expression `* + 7d` to set the time stamp to exactly seven days from the trigger time. The `WeeklyGoal` attribute will have a time stamp that is one week in the future (that is, the following Sunday at 12:00 a.m.).

Expression analyses

Expression analyses calculate one or more output values from specified functions, operators, and input values. You specify the inputs and calculations for these analyses in expressions. You can map any output that an expression calculates to an attribute. You must map at least one output to an attribute.

You can map the output to an existing attribute or you can create a new attribute when you map the output. If you create new attribute when you map the output (from the **Analyses** tab in PI System Explorer), you must choose whether or not to save the output history; the system creates the appropriate type of attribute:

- If you choose to save output history, the system creates an attribute with a PI point data reference and creates a PI point in the specified PI Data Archive. With the output saved in a PI point, you can use visualization tools to see the trend associated with the calculation, and you can retrieve the value for any time that the analysis performed the calculation. The analysis calculates and stores a value as specified by the analysis schedule.
- If you choose not to save output history, PI System Explorer creates an attribute with an analysis data reference. An analysis only calculates the value for expressions mapped to attributes with an analysis data reference when requested. For best performance, map output attributes to attributes with analysis data references if you only need to know the most recent value and do not need previous values.

Topics in this section

- Create an expression analysis template
- Create an expression analysis

Create an expression analysis template

In PI System Explorer, you can create an expression analysis template for an element template. In the template, you can create an expression for each calculation. To record the value that an expression calculates, map the output to an attribute template. You can create new attribute templates from the expression-analysis template.

Before you start

Determine output values you want to calculate and save. Identify the functions, operators, and input attributes needed for the calculations.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Expression**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.
7. Optional. To temporarily evaluate expressions in the analysis template based on values of a particular element, click the **Select an example element** link and select an element based on the current element template. If no elements have been created from the current element template, you must create an element first.

   Templates have no concrete data associated with their attributes. You cannot evaluate an expression in an analysis template, unless you borrow attribute values from a specific element.

   PI System Explorer sets **Example Element** to the element you selected.
8. Specify one or more expressions for the analysis.

   Enter each expression in a row. An expression specifies inputs and calculations. See Expressions for analyses.
9. Map at least one expression to an output attribute template.

a. In the **Output Attribute** column for the expression, click **Map**.

b. Specify the attribute template.

To ...	Do this ...
Map the output to an existing attribute template	Click the name of the attribute template. Select an attribute template with a PI point data reference to store the history of the calculation. For best performance when you do not require history, use an attribute template with an analysis data reference.
Map the output to a new attribute template	a. Click **New Attribute Template** to open the Attribute Template Properties window. b. Indicate whether the attribute should save the history of the output: ▪ Click **Yes** to create an attribute with a PI point data reference. This attribute saves the history of the output. ▪ Click **No** to create an attribute with an analysis data reference. This attribute calculates a new value when requested. c. Configure the attribute template to create: ▪ **Name** Name of created attribute. ▪ **Description** Optional text that describes the attribute. ▪ **Data Server** For attributes that save output history, the PI Data Archive server that stores the PI point data reference. By default, this is set to `%Server%`, which sets to the default PI Data Archive server. ▪ **Value Type** The data type the attribute stores. After creating the attribute template, you can refine its definition from the **Attribute Templates** tab. d. Click **OK** to create the attribute template.

10. Optional. If you specified an example element, click **Evaluate** to verify that output values of individual expressions.

11. Specify scheduling for the analysis. See Analysis scheduling.

12. Optional. If you specified an example element, verify the analysis by reviewing the results produced with historical data:

    a. In the analysis list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

13. To apply changes and save your work locally, click ✔ on the toolbar.
14. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an expression analysis

In PI System Explorer, you can create a unique expression analysis for an individual element. In the analysis, you can create an expression for each calculation. To record the value that an expression calculates, map the output to an attribute. You can create new attributes as you configure an analysis.

Before you start

Determine output values you want to calculate and save. Identify the functions, operators, and input attributes needed for the calculations.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Expression**.
6. Specify one or more expressions for the analysis.

   Enter each expression in a row. An expression specifies inputs and calculations. See Expressions for analyses.
7. Map at least one expression to an output attribute:

   a. In the **Output Attribute** column for the expression, click **Map**.

   b. Specify the attribute.

To ...	Do this ...
Map the output to an existing attribute	Click the name of the attribute. Select an attribute with a PI point data reference to store the history of the calculation. For best performance when you do not require history, use an attribute with an analysis data reference.
Map the output to a new attribute	a. Click **New Attribute** to open the Attribute Properties window. b. Indicate whether the attribute should save the history of the output: ▪ Click **Yes** to create an attribute with a PI point data reference. This attribute saves the history of the output. ▪ Click **No** to create an attribute with an analysis data reference. This attribute calculates a new value when requested. c. Configure the attribute: ▪ **Name** Name of created attribute. ▪ **Description** Optional text that describes the attribute. ▪ **Data Server** For attributes that save output history, the PI Data Archive server that stores the PI point data reference. By default, this is set to `%Server%`, which sets to the default PI Data Archive server. ▪ **Value Type** The data type the attribute stores. After creating the attribute, you can refine its definition from the **Attributes** tab. d. Click **OK** to create the attribute.

8. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times.
9. Specify scheduling for the analysis. See Analysis scheduling.
10. Optional. To verify the analysis, review the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
11. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.
12. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Rollup analyses

A rollup analysis calculates statistical values such as sum and average for selected attributes associated with an element. For example, a rollup analysis on a factory element might use temperature attribute values for all pumps in the factory to calculate their average temperature.

You can choose either of two sources for attributes to include in a rollup analysis:

- **Attributes of the element**

  For example, you want to verify that the level of a tank is constant by checking that all inflows and outflow sum to zero. From the list of the tank element's attributes, select its inflow and outflow attributes for a summation calculation. A rollup using an element's own attributes is also known as an aggregation.

- **Attributes of the element's child elements**

  For example, you want to calculate average energy consumption for a group of pumps in a refinery. To do that, you create a roll-up analysis on the parent element (the refinery) that averages energy consumption attributes from its child elements (pumps).

Use selection criteria to select attributes for rollup. You can select attributes that match attribute name text you enter or that are associated with a category or element template you specify.

With PI Server 2015 R2 and later versions, you can select child attributes for rollup analyses. For more information, see Create a rollup analysis.

You must specify one or more functions to calculate statistics on the selected attributes in the rollup (see Rollup analysis functions). Be sure each output is mapped to an output attribute.

Be aware that:

- You should select attributes carefully to obtain sensible results. Avoid selecting a set with mixed units of measure (UOM; such as temperature and mass) or value types (such as numeric values and time stamps). As long as the inputs and outputs have different units within the same UOM class, the system makes reasonable effort to convert the input units to that of the output for calculation.
- You can add or delete elements or attributes in your hierarchy without the need to update roll-up analyses. Because a rollup identifies input attributes each time it is executed, it automatically includes any new attributes that meet its selection criteria.
- Some selected attributes (such as those with string values) will not participate in rollup calculations, which operate only on numeric or `DateTime` values.

Topics in this section

- Rollup analysis functions
- Create a rollup analysis template
- Create a rollup analysis

Rollup analysis functions

You can specify one or more of the functions listed below to calculate statistics on the input attributes in a rollup analysis.

> **Note:**
> Input value types numeric, `DateTime` and `Enumeration value` are supported for **`Minimum`**, **`Maximum`**, and **`Count`**. **`Average`** takes numeric and `DateTime` value types. Input values for all other functions must be numeric value type.

Function	Description	Supported data types
Sum	The sum of values from selected attributes.	Numeric
Average	The average (mean) of values from two or more selected attributes.	Numeric, `DateTime`
Minimum	The minimum value from selected attributes.	Numeric, `DateTime`, `Enumeration value`
Maximum	The maximum value from selected attributes.	Numeric, `DateTime`, `Enumeration value`
Count	The number of attributes actually used in rollup calculations, which can differ from the total number of selected attributes. For example, an attribute containing a string value (such as a part name) might meet the selection criteria but would not participate in rollup calculations and would be excluded for **`Count`**.	Numeric, `DateTime`, `Enumeration value`
Median	The middle value of three or more values from selected attributes; for an even-numbered group of values, the average of the two middle values. An equal number of values fall above and below the median.	Numeric
Population standard deviation	The population standard deviation, which is calculated using the entire population. Useful when all values in a population are available to include as inputs for the calculation.	Numeric
Sample standard deviation	The sample standard deviation, which is calculated using values from a sample of a population. Useful to provide an estimate of the population standard deviation when it is impractical or impossible to include all population values in the calculation.	Numeric

Create a rollup analysis template

In PI System Explorer, you can create a rollup analysis template for a selected element template. You select input values for the rollup either from attributes of the selected element template or from the attributes of a child element template (whose parent element is derived from the element template). You can choose an example child element and use its attributes to help you develop a rollup analysis template.

Before you start

Identify the statistical functions you want the rollup to calculate and where you want to save the results. You can map results from a function to an existing attribute of the element template, or you can create a new output attribute when you configure an analysis template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Rollup**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.
7. Under **Rollup attributes from**, indicate the source of input values:
   - Click **Child elements of *Element Template Name*** to roll up attributes of child elements of an element based on the current element template.
   - Click **This element - *Element Template Name*** to roll up attributes of an element based on the current element template.
8. Select an example element to provide candidate attributes:
   a. Click the **Select an example element** link to open a window that shows elements derived from the selected element template.
   b. Select an element and click **OK**.
9. Select the attributes to include in the rollup:

If rollup attributes come from:	Then:
Child elements	a. From the **Sample Child Element** list, select the element from which you want to view attributes. The table shows attributes of the selected element. To improve performance, the table only shows a limited number of attributes. You can click the **Show more attributes** link below the table to show additional attributes from the selected element. b. To the left of the table, specify criteria that select attributes to roll up: ▪ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp", such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ▪ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ▪ **Attribute Category** Select a category to limit selected attributes to those in that category. ▪ **Element Category** Select an element category to limit selected attributes to attributes that have parent elements in that category. ▪ **Element Template** Select an element template to limit selected attributes to attributes that have parent elements based on that template. **Note:** The criteria apply to all attributes of any child element, not just those shown in the table. You can specify criteria that select attributes not shown in the table. For example, if you set **Sample Child Element** to an element from category `CatX` but set **Element Category** to `CatY`, you will select none of the attributes in the table but the analysis will still include the attributes in the rollup.

This element	To the left of the table, specify criteria that select attributes to roll up: ◦ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp", such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ◦ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ◦ **Attribute Category** Select a category to limit selected attributes to that category.

The table shows a check mark next to any displayed attribute included in the rollup.

10. In the functions table, specify the rollup calculations and output:
    a. Select the check boxes next to any statistical function you want the rollup to calculate.
    b. Click **Map** to store results from a calculation to an output attribute.

       You can map the output to an existing attribute or map the output to a new output attribute.
    c. Optional. At the top of the table, click **Evaluate** to verify the output values of the selected functions.
11. Specify scheduling for the analysis. See Analysis scheduling.
12. Optional. To verify the analysis, examine the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
13. To apply changes and save your work locally, click ✔ on the toolbar.
14. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create a rollup analysis

In PI System Explorer, you create a rollup analysis for a selected element. You select input attributes for the rollup either from attributes of the selected element or from attributes of child elements.

Before you start

Identify the statistical functions you want the rollup to calculate and where you want to save the results. You can map results from a function to an existing attribute of the element, or you can create a new output attribute when you configure the analysis.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Rollup**
6. Under **Rollup attributes from**, indicate the source of input values:
   - Click **Child elements of *Element Name*** to roll up attributes of a child element.
   - Click **This element - *Element Name*** to roll up attributes of the current element.
7. Select the attributes to include in the rollup:

If rollup attributes come from ...	Do this ...
Child elements	a. From the **Sample Child Element** list, select the element from which you want to view attributes. The table shows attributes of the selected element. To improve performance, the table only shows a limited number of attributes. You can click the **Show more attributes** link below the table to show additional attributes from the selected element. b. To the left of the table, specify criteria that select attributes to roll up: ▪ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp," such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ▪ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ▪ **Attribute Category** Select a category to limit selected attributes to those in that category. ▪ **Element Category** Select an element category to limit selected attributes to attributes that have parent elements in that category. ▪ **Element Template** Select an element template to limit selected attributes to attributes that have parent elements based on that template. **Note:** The criteria apply to all attributes of the selected child element, not just those shown in the table. You can specify criteria that select attributes not shown in the table. For example, if you set **Sample Child Element** to an element from category `CatX` but set **Element Category** to `CatY`, you will select none of the attributes in the table but the analysis will still include the attributes in the rollup.

This element	To the left of the table, specify criteria that select attributes to roll up:
	◦ **Attribute Name** Type an entire attribute name or part of a name with wildcard characters. For example, type `temp*` to select all attributes that begin with "temp," such as `Temperature` and `Temp1`. **Note:** The selection of attributes depends on whether **`Attribute Level`** is set to "Root Level" or "Child Level". ◦ **Attribute Level** Select **Root Level** to choose from top-level attributes or **Child Level** for child attributes. You cannot choose both child attributes and top-level attributes in a single roll-up analysis. ◦ **Attribute Category** Select a category to limit selected attributes to that category.

The table shows a check mark next to any displayed attribute included in the rollup.

8. In the functions table, specify the rollup calculations and output:
   a. Select the check boxes next to any statistical function you want the rollup to calculate.
   b. Click **Map** to store results from a calculation to an output attribute.

      You can map the output to an existing attribute or map the output to a new output attribute.
   c. Optional. At the top of the table, click **Evaluate** to verify the output values of the selected functions.
9. Specify scheduling for the analysis. See Analysis scheduling.
10. Optional. To verify the analysis, review the results produced with historical data:
    a. In the analysis list, right-click the analysis and then click **Preview Results**.
    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.
11. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.
12. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Event frame generation analyses

An event frame generation analysis specifies the conditions to start and end event frames automatically.

This type of analysis includes either one or two expressions. When a single condition triggers both the start and the end of an event frame, only a start trigger expression is needed, depicted by the **StartTrigger1** field. For example, a temperature value rising above a threshold might start an event frame and end it when the value returns below the threshold. When the start and end conditions are different, an **EndTrigger** expression is also needed. Because they test start and end conditions, expressions in event frame generation analyses must evaluate to true or false. For more information, see Start and end trigger conditions.

You can write up expressions to be used as start or end triggers. If your expression gets very complex, you can break it into a group of smaller expressions and assign them to variables. For more information, see Expression simplification with refactoring.

Beginning with the 2018 release, a new way of event frame generation is possible where you do not have to set an explicit trigger. For more information, see Implicit modes of event frame generation in asset analytics.

When configuring event frame generation analyses, you can add expressions to **Outputs at Close** group to write outputs back to event frame attributes. Depending on whether you want to save the history of the output, you can select either attributes configured with a PI point data reference or static event frame attributes. When saving output history to event frame attributes configured as a PI Point data reference, the attribute must be mapped to a corresponding PI Point data reference attribute of the referenced element.

Note:

- Outputs from root cause event frames are only written to static attributes.
- Parent event frames with multiple child event frames will also write only to static attributes. For more information on child event frame generation, see Child event frame generation with multiple start triggers.

In your expressions, you can use inputs that come from assets to write outputs to event frame attributes at the close of an event frame. You can also use the **`EventFrame`** function to access the start time, end time and duration of the event frame. For more information, see expression function EventFrame.

An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

Note:

- Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
- An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
- Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff%`) will be used in the absence of a name or if the event frame name evaluates to a null value.

You can create multiple start triggers for your analysis and specify different Boolean conditions to trigger each event frame. For more information on multiple start triggers and child event frame generation, see Child event frame generation with multiple start triggers.

> **Note:**
> If you change the name of a start trigger from the default name, you must change all the start triggers and use unique names for each trigger.

A spike in input data can trigger the start of an unwanted event frame. To counter the effects of data spikes, you can require that the start trigger remain true for a set time interval before creating an event frame. Enter a value in the **True for** field to specify the time interval.

You can also specify a Severity level for the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical.

An event frame generation analysis is based on an event frame template, which specifies the attributes for the generated event frames. Before you create an event frame generation analysis, be sure an appropriate event frame template is available.

Event frames usually include a referenced element collection. The element associated with an event frame generation analysis becomes the primary reference element for its generated event frames. For more information, see Primary referenced element. Other elements can be referenced using relative paths based on the primary referenced element, as described in Data references to attributes in other elements.

For some events, such as a forced shutdown, you may want to evaluate the conditions leading up to the event. To do that, you can have PI Analysis Service generate root cause event frames. When configured, every generated event frame includes a child root cause event frame that captures attributes for a specified time interval just before the start of the generated event frame. For more information on generating a root cause event frame, see step 11 on Create an event frame generation analysis template.

Video

For information on how to set up event frame generation analyses, watch this video:

https://www.youtube.com/watch?v=iNYe_YZzj58&rel=0

Topics in this section

- Implicit modes of event frame generation in asset analytics
- Create an event frame generation analysis template
- Create an event frame generation analysis
- Start and end trigger conditions
- Child event frame generation with multiple start triggers

Implicit modes of event frame generation in asset analytics

Beginning with the 2018 release, users of asset analytics in PI AF have the option to select different modes of event frame generation. Prior to 2018, users had to explicitly set a start trigger (now called Explicit Trigger). In 2018 or later versions of PI AF, you can choose among the three implicit modes in addition to an explicit trigger: pulse, step and step continuous. After having selected Event Frame Generation as your analysis type, select one of the following as **Generation Mode**:

- Pulse: A transition from zeroth state to any other state starts an event frame, and transition to the zeroth state ends the event frame. You have an option to generate a root cause event frame in **Advanced Event Frame Settings**.
- Step: All transitions would close the open event frame and start a new event frame unless the transition is to a zeroth state
- Step Continuous: All transitions would close the open event frame and start a new event frame

Note that you must select a triggering attribute that is either integer, string or enumeration value for these event frames. For pulse and step event frames, you will need to select an attribute as a trigger and set zeroth state. Zeroth state is not supported for step continuous event frames. Zeroth state determines when the event frame should be closed. You can select zeroth states from digital state or enumeration sets. For more information, see PI EFGen topic "Event start and end detection" in Live Library (https://livelibrary.osisoft.com).

Create an event frame generation analysis template

In PI System Explorer, you can create an event frame generation analysis template for an element template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Event Frame Generation**.
6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.

7. Optional. To temporarily evaluate expressions in the analysis template based on values of a particular element, click the **Select an example element** link and select an element based on the current element template. If no elements have been created from the current element template, you must create an element first.

   Templates have no concrete data associated with their attributes. You cannot evaluate an expression in an analysis template, unless you borrow attribute values from a specific element.

   PI System Explorer sets **Example Element** to the element you selected.

8. From the **Event Frame Template** list, select the template for the event frame that this analysis generates.

   To create an event frame template, see Create event frame templates. An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

   Note:

   - Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
   - An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
   - Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff %`) will be used in the absence of a name or if the event frame name evaluates to a null value.

9. Specify the necessary expressions for the analysis:

   a. In the **StartTrigger1** row, enter the Boolean expression that specifies the condition that starts an event frame.

   b. Optional. Change the default name of the start trigger.

      Note:
      If you change the name of a start trigger from the default name, you must change the names of all start triggers and use unique names for each trigger.

   c. Optional. In the **True for** field, enter the amount of time that the start-trigger condition must be true before the analysis starts an event frame.

      Note:
      Specify a value in the **True for** field to reduce unwanted event frames caused by momentary fluctuations in input data.

   d. Optional. In the **Severity** field, enter the severity of the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical. If you have defined multiple start triggers, the one with the highest severity will be the effective start trigger; if all start triggers have the same severity, the first start trigger in the list will be the effective start trigger.

   e. Add multiple start triggers by clicking on **Add a new start trigger**. For each start trigger, enter a Boolean expression, specify a value for **True for** and select a value for **Severity**.

> **Note:**
> If you have changed the default name of a start trigger, make sure to assign unique names to all the start triggers.

f. Optional. If a different condition ends the event frame, enter the expression in the **EndTrigger** row. Otherwise, the event frame will close when the start trigger condition is no longer true.

g. Optional. Click **Add a new variable** to insert a new expression row where you can specify a variable and expression for use in one of the trigger expressions.

h. Optional. Click **Add a new output expression** to insert a new expression to write outputs to event frame attributes. You can map the output as static (doesn't save history) or PI point (saves history) attribute. When mapping to PI point attribute, make sure that it is mapped back to the attribute on the asset.

10. Optional. If you specified an example element, click **Evaluate** to verify that output values of individual expressions.

11. Optional. Generate a root cause event frame for every event frame that the analysis generates. A root cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

> **Note:**
> Outputs from root cause event frames are only written to static attributes.

a. From the Advanced Event Frame Settings window, select **Generate child root cause event frame before parent event frame starts**.

b. In the **Duration** field, enter the length of the root cause event frame.

The root cause event frame starts this long before the generated parent event.

12. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger name is saved.

b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger expression is saved.

The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

13. Specify scheduling for the analysis. See Analysis scheduling.

14. Optional. If you specified an example element, verify the analysis by reviewing the results produced with historical data:

a. In the analysis list, right-click the analysis and then click **Preview Results**.

b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

15. To apply changes and save your work locally, click ✔ on the toolbar.

16. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an event frame generation analysis

In PI System Explorer, you can create an event frame generation analysis for an individual element.

Before you start

Check that an event-frame template is available for the analysis.

Procedure

1. In the navigator, click **Elements**.
2. In the Elements browser, select the element where you want to create the analysis.
3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.
4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **Event Frame Generation**.
6. From the **Event Frame Template** list, select the template for the event frame that this analysis generates.

   To create an event frame template, see Create event frame templates. An event frame naming pattern typically includes substitution parameters. For more information on naming pattern, see Naming patterns.

Note:

- Parameters `%Endtime%` and `%Duration%` in an event frame naming pattern will be evaluated at the end of an event frame when the event frame closes.
- An attribute value may update at any given time. To avoid confusion, substitution parameter `%@Attribute%` (which resolves to a value of an attribute) in an event frame naming pattern will always be (re-)evaluated in the context of a start of an event frame.
- Default naming pattern (`%ANALYSIS% %STARTTIME:yyyy-MM-dd HH:mm:ss.fff %`) will be used in the absence of a name or if the event frame name evaluates to a null value.

7. Specify the necessary expressions for the analysis:
   a. In the **StartTrigger1** row, enter the Boolean expression that specifies the condition that starts an event frame.
   b. Optional. Change the default name of the start trigger.

      Note:
      If you change the name of a start trigger from the default name, you must change all the start triggers and use unique names for each trigger.

   c. Optional. In the **True for** field, enter the amount of time that the start-trigger condition must be true before the analysis starts an event frame.

      Note:
      Specify a value in the **True for** field to reduce unwanted event frames caused by momentary fluctuations in input data.

   d. Optional. In the **Severity** field, enter the severity of the trigger. The choices for severity are: None, Information, Warning, Minor, Major, and Critical. If you have defined multiple start triggers, the one with the highest severity will be the effective start trigger; if all start triggers have the same severity, the first start trigger in the list will be the effective start trigger.
   e. Add multiple start triggers by clicking on **Add a new start trigger**. For each start trigger, enter a Boolean expression, specify a value for **True for** and select a value for **Severity**.

      Note:
      If you have changed the default name of a start trigger, make sure to assign unique names to all the start triggers.

   f. Optional. If a different condition ends the event frame, enter the expression in the **EndTrigger** row. Otherwise, the event frame will close when the start trigger condition is no longer true.
   g. Optional. Click **Add a new variable** to insert a new expression row where you can specify a variable and expression for use in one of the trigger expressions.
   h. Optional. Click **Add a new output expression** to insert a new expression to write outputs to event frame attributes. You can map the output as static (doesn't save history) or PI point (saves history) attribute. When mapping to PI point attribute, make sure that it is mapped back to the attribute on the asset.

   See Expressions for analyses for more information.

8. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times.

9. Optional. Generate a root cause event frame for every event frame that the analysis generates. A root cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

   Note:
   Outputs from root cause event frames are only written to static attributes.

   a. From the Advanced Event Frame Settings window, select **Generate child root cause event frame before parent event frame starts** .

   b. In the **Duration** field, enter the length of the root cause event frame.

      The root cause event frame starts this long before the generated parent event.

      Note:
      The root cause event frame is created only for the first instance, not for each time that the trigger is `True`.

10. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

    Note:
    We recommend that you provide meaningful names to your analysis start triggers while saving them to an event frame attribute.

    a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger name is saved.

    b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute (template) or create a new attribute template on the event frame template to which the start trigger expression is saved.

       The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

11. Specify scheduling for the analysis. See Analysis scheduling.

12. Optional. To verify the analysis, review the results produced with historical data:

    a. In the analysis list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

13. To apply changes and save your work locally, click ✔ on the toolbar. This does not run the analysis.

14. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Start and end trigger conditions

An event frame generation analysis starts an event frame when the start trigger condition evaluates to true. Because a random spike in an input value can make the condition true, noisy data can lead to the creation of many unwanted event frames.

To counter the effects of data spikes, you can require that the start trigger condition remain true for a set time interval before starting an event frame. Enter a value in the **True for** field to specify that time interval.

The event frame generation analysis will not create a new event frame until the expression in the start trigger has been true for the duration specified in **True for** field. The start time for the event frame is the time that the start trigger condition first evaluated to true.

If the start trigger condition evaluates to false before it reaches the time interval specified in **True for** field, no event frame is created.

The **True for** value has no effect on event-frame end time.

The event frame will close when the start trigger condition is false. Configuring an end trigger is optional. If both the start and end trigger conditions are configured, the event frame will end when the end trigger condition is met. This is regardless of whether the start trigger condition still holds true.

Tip:

- If using both start and end triggers, make sure the expressions never evaluate to TRUE at the same time since this may lead to event frames with zero second durations.
- Try to configure your event frames to use only a start trigger expression.

Child event frame generation with multiple start triggers

PI Analysis Service can capture child event frames. Child event frames are useful when you need to send escalating notifications or perform an in-depth analysis of the past events. This topic discusses the following:

- Video that shows you how to configure event frames with multiple start triggers
- Effective start triggers
- Example of child event frame generation

Note:
This topic is only applicable to the explicit trigger.

Video

For information on setting up event frames with multiple start triggers and severities, watch this video:

https://www.youtube.com/watch?v=R15X4fYYxAQ&rel=0

Effective start triggers

Event frame generation analysis supports multiple start triggers. When triggering an analysis, the *effective start trigger* is determined by taking the following three things into account: trigger evaluation, severity and order of the triggers.

- **Evaluation**

  If only one start trigger is `True`, then that is the effective start trigger. If multiple start triggers evaluate to `True`, then the analysis must take severity into account.

- **Severity**

  If the multiple start triggers evaluate to `True`, the one with the highest severity level is the effective start trigger. For example, if there are two start triggers, `StartTrigger1` with severity `Major` and `StartTrigger2` with severity `Critical`, and they both evaluate to `True` at the same point in time, then `StartTrigger2` is considered the effective trigger as it has the higher severity.

  > **Note:**
  > Severity levels in decreasing order of magnitude are: `Critical`, `Major`, `Minor`, `Warning`, `Information` and `None` (default).

- **Order**

  If multiple start triggers with the same severity evaluate to `True`, then the start trigger defined first is the effective start trigger.

In the end, there is only one effective start trigger.

Child event frame generation due to change in the effective start trigger

An analysis with multiple start triggers may automatically create child event frames. For example, an event frame generation analysis would complete the following steps:

1. A start trigger evaluates to `True` and there is no open event frame. The analysis starts an event frame.
2. In a subsequent evaluation, while the event frame is still open, the effective start trigger does not change and continues to evaluate to `True`. Then the event frame will remain open.
3. Another evaluation causes the effective start trigger to be different than step 1. This causes the analysis to do the following:
   a. Start a new parent event frame with the same start time as the original event frame.
   b. Make the original event frame a child of this new parent event frame.
   c. Close the original event frame (now a child event frame).
   d. Create a new child event frame with the new effective start trigger.
4. Another evaluation does not cause the effective start trigger to change. The analysis keeps both the child and the parent event frames open.
5. Another evaluation causes the effective start trigger to change again. Then:
   a. The child event frame created in 3d closes.
   b. Yet another child event frame is created by this new effective start trigger.
6. Another evaluation causes all of the start triggers to be `False`. Then:
   a. Child event frame created in 5b closes.
   b. Parent event frame created in 3a closes.

SQC analyses

Using PI System Explorer, you can create SQC analyses that use standard Statistical Quality Control (SQC) tests to determine if the value of a PI AF attribute lies within user-specified

margins of error. Further, you can specify either attributes or event frames as outputs of your SQC analysis.

The following SQC pattern tests may be selected:

- **Outside Control:**

  Counts the number of samples outside the control limit on one side of the center line.

- **Outside 2 Sigma:**

  Evaluates the sample against a limit drawn 2/3 of the way between the center line and the control limit.

- **Outside 1 Sigma:**

  Evaluates the sample against a limit drawn 1/3 of the way between the center line and the control limit.

- **One Side of Center Line:**

  Counts the number of samples on one side of the center line.

- **Stratification:**

  Counts the number of samples that fall within the upper and lower One Sigma limits on both sides of the center line.

- **Mixture:**

  Counts the number of samples that fall outside the upper and the lower One Sigma limits on both sides of the center line.

- **Trend:**

  Counts the number of samples which are increasing or decreasing in a monotonic sequence.

Create an SQC analysis template

In PI System Explorer, you can create an SQC analysis template for an element template.

Procedure

1. In the navigator, click **Library**.
2. In the Library browser, select the element template where you want to create the analysis template.
3. In the viewer, click the **Analysis Templates** tab.

   The tab lists any analysis templates already defined for the element template.
4. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.

- **Description**

  Optional text that describes the analysis.

- **Categories**

  Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.

- **Analysis Type**

  Click **SQC**.

6. Optional. Clear the **Enable analyses when created from template** check box so that the analysis is created in `Disabled` status.

   By default, a new analysis will be created with `Enabled` status. You must clear the option if you do not want to start the analysis when created.

7. Optional. Select an example element.

8. Provide the inputs for the SQC pattern test. All these inputs should be attribute templates previously defined in your PI System.

   a. In the **Source** field, select the PI AF attribute template that provides the data for the pattern test.

   b. In the **Upper Control Limit** and **Lower Control Limit** fields, select the PI AF attribute template that provides the data for the largest and smallest limits on the dimension of the data being tested.

   c. In the **Center Line** field, select the PI AF attribute template that provides the normal or ideal value of the data.

9. Choose the output of your analysis as either **Event Frame** or **AF attribute**. If you choose **Event Frame**, select an event frame template. If you choose **AF attribute**, you can either choose from a list, or create a new attribute template. SQC Analysis rule automatically creates an enumeration set named **SQC Enumeration**. This enumeration set has the same values used in PI Data Archive SQC DigitalState set so that continuity with existing PI Data Archive is preserved. **SQC Enumeration** is created when: 1) you map the output to a new **AF attribute**, 2) this or another enumeration set with the required name and values does not already exist and 3) you have the right to create enumeration sets. **AF attribute** output defaults to a string attribute if the aforementioned conditions are not met.

10. Optional. Generate a root-cause event frame for every event frame that the analysis generates. A root-cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.

    a. From the **Advanced Event Frame Settings** menu, select **Generate Child Root Cause Event Frame Before Parent Event Frame Starts** .

    b. In the **Duration** field, enter the length of the root-cause event frame.

       The root-cause event frame starts this long before the generated parent event.

       > **Note:**
       > The root-cause event frame is created only for the first instance, not for each time that the trigger is `True`.

11. Optional. Map a start trigger name and start trigger expression to an event frame attribute.

a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute template on the event frame template to which the start trigger name is saved.

b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute template on the event frame template to which the start trigger expression is saved.

The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.

12. Enter SQC pattern test information.

a. Select the check box for each SQC pattern test that you wish to apply to your data.

b. In the **X of Y** column, select appropriate numerical values for X and Y for each selected pattern test that requires this input. Specify the number of values X out of a number of sampled values Y that can be above or below the corresponding pattern test limits.

Patterns (X and Y values) suggested by Western Electric are shown by default.

> **Note:**
> X may not be greater than Y.

c. Select the **Limit** as **Above**, **Below** or **Both**, for each selected pattern test. By default, tests are applied on both sides of the center line.

13. Optional. Select the **Clear on Control Limit Change** check box.

With **Clear on Control Limit Change** selected, if any of the control limits changes, asset analytics resets the state of a running SQC calculation, and the SQC calculation is restarted with the changed values. In addition, associated open event frames are closed before the calculation is resumed.

With **Clear on Control Limit Change** not selected, the calculation simply proceeds using the changed values for the control limits.

14. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times. If all selected SQC tests pass, the **Output Value** field displays "Normal". If some tests fail, the **Output Value** field shows the highest priority test that failed. Changing any of the SQC inputs will clear the **Output Value** field.

15. Specify scheduling for the analysis. See Analysis scheduling.

16. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create an SQC analysis

Before you start

Identify the Statistical Quality Control (SQC) pattern tests that you wish to use. See SQC analyses for more information.

Procedure

1. In the navigator, click **Elements**.

2. In the Elements browser, select the element where you want to create the analysis.

3. In the viewer, click the **Analyses** tab.

   The tab lists any analyses already defined for the element.

4. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.

5. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis.
   - **Description**

     Optional text that describes the analysis.
   - **Categories**

     Optional category that you can assign to the analysis. Click the list and select the check box next to categories you want to assign or click **New** to create a new category.
   - **Analysis Type**

     Click **SQC**.

6. Provide the inputs for the SQC pattern test. All these inputs should be attributes or attribute templates previously defined in your PI System.
   a. In the **Source** field, select the PI AF attribute or attribute template that provides the data for the pattern test.
   b. In the **Upper Control Limit** and **Lower Control Limit** fields, select the PI AF attribute or attribute template that provides the largest and smallest limits on the dimension of the data being tested.
   c. In the **Center Line** field, select the PI AF attribute or attribute template that provides the normal or ideal value of the data.

7. Choose the output of your analysis as either **Event Frame** or **AF attribute**. If you choose **Event Frame**, select an event frame template. If you choose **AF attribute**, you can either choose from a list, or create a new attribute. SQC Analysis rule automatically creates an enumeration set named **SQC Enumeration**. This enumeration set has the same values used in PI Data Archive SQC DigitalState set so that continuity with existing PI Data Archive is preserved. **SQC Enumeration** is created when: 1) you map the output to a new **AF attribute**, 2) this or another enumeration set with the required name and values does not already exist and 3) you have the right to create enumeration sets. **AF attribute** output defaults to a string attribute if the aforementioned conditions are not met.

8. Optional. Generate a root-cause event frame for every event frame that the analysis generates. A root-cause event frame captures asset data, which can help you analyze conditions just before the start of the event frame.
   a. From the **Advanced Event Frame Settings** menu, select **Generate Child Root Cause Event Frame Before Parent Event Frame Starts** .
   b. In the **Duration** field, enter the length of the root-cause event frame.

      The root-cause event frame starts this long before the generated parent event.

> **Note:**
> The root-cause event frame s created only for the first instance, not for each time that the trigger is `True`.

9. Optional. Map a start trigger name and start trigger expression to an event frame attribute.
   a. From the Advanced Event Frame Settings window, select **Save start trigger name to event frame attribute** and then select an attribute or attribute template on the event frame template to which the start trigger name is saved.
   b. From the Advanced Event Frame Settings window, select **Save start trigger expression to event frame attribute** and then select an attribute or attribute template on the event frame template to which the start trigger expression is saved.

      The start trigger name and start trigger expression can now be seen on generated event frames, on the selected attributes.
10. Enter SQC pattern test information.
    a. Select the checkbox for each SQC pattern test that you wish to apply to your data.
    b. In the **X of Y** column, select appropriate numerical values for X and Y for each selected pattern test that requires this input. Specify the number of values X out of a number of sampled values Y that can be above or below the corresponding pattern test limits.

       Patterns (X and Y values) suggested by Western Electric are shown by default.

       > **Note:**
       > X may not be greater than Y

    c. Select the **Limit** as **Above**, **Below** or **Both**, for each selected pattern test. By default, tests are applied on both sides of the center line.
11. Optional. Select the **ClearOnControlLimitChange** checkbox. With **ClearOnControlLimitChange** selected, if any of the control limits changes, asset analytics resets the state of a running SQC calculation, and the SQC calculation is restarted with the changed values. In addition, associated open event frames are closed before the calculation is resumed.

    With **ClearOnControlLimitChange** not selected, the calculation simply proceeds using the changed values for the control limits.
12. Optional. Click **Evaluate** to verify the output values at both **Value at Last Trigger** and **Value at Evaluation** times. If all selected SQC tests pass, the **Output Value** field displays "Normal". If some tests fail, the **Output Value** field shows the highest priority test that failed. Changing any of the SQC inputs will clear the **Output Value** field.
13. Click **Check In** on the toolbar.

    PI System Explorer saves and checks in your analysis, and then starts running the analysis.

Sample analyses

The sample analyses in this section perform the following calculations in the same PI AF database:

1. Uses an expression type analysis to track the deviation from target efficiency for the asset and processes represented by the element, `Ethylene`.
2. Creates a template to generate a set of rollup analyses that aggregate the total power draw of all processes at a refinery. The element template from which the element for each refinery was created is called `Refinery`.
3. Creates event frames to capture data any time the 15-minute running-average efficiency for the element, `Ethylene`, drops below 90%.

Build an expression analysis to study efficiency deviation

This example demonstrates how to build an expression type analysis. The analysis tracks how the efficiency of a process deviates from the target efficiency.

First, we'll calculate the hourly average efficiency, then determine how much it varies from the target efficiency.

This example creates a new PI point to store the output data (the deviation from target efficiency, as it varies over time). When you create your own analyses, OSIsoft recommends you create the PI points you'll need for output data before you start to create the analysis.

Before you start

Create an element named `Ethylene` that has an attribute named `Efficiency` that contains a PI point data reference.

Procedure

1. Under **Elements**, select the element for which you want to build the analysis, such as **Ethylene**.

2. Click the **Analyses** tab.

3. Create a new analysis for the element:
   - If no analyses exist, click **Create a new analysis** to create the first one.
   - Otherwise, click **New Analysis** to create a new one.
4. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis. For example, `EfficiencyCalc`.
   - **Analysis Type**

     Click **Expression**. The expression analysis is the simplest type of analysis; it contains one or more expressions plus scheduling information. For more information, see Expression analyses.
5. In the expressions table, create a variable to hold the constant value of the target efficiency:
   a. In the **Name** column, type `Target`.

   b. In the **Expression** column, enter `90`.

   This sets this variable to the constant value of 90.
6. Create a second variable to calculate the hourly average efficiency:
   a. Click **Add a new expression**.
   b. Name the variable `HourlyEfficiency`.
   c. In the **Expression** column, enter the following performance equation syntax:
   ```
   TagAvg('Efficiency', '*-1h', '*')
   ```

   This expression calculates the average value of the attribute called `Efficiency` over the past hour.
7. Create a third variable to calculate how much the hourly efficiency deviates from the target of 90:
   a. Click **Add a new expression**.
   b. Name the variable `Deviation`.
   c. In the **Expression** column, enter the following expression:
   ```
   HourlyEfficiency - Target
   ```
8. Save the output from the `Deviation` variable to a PI point:
   a. In the **Output Attribute** column, click **Map**.
   b. Click **New Attribute** to create a new PI AF attribute.
   c. To define the attribute as a PI point data reference, click **Yes** to save output history.
   d. In the **Name** field, enter the attribute name, such as `EfficiencyDifference`.
   e. Click **OK** to create the attribute.

9. Specify scheduling for the analysis:
   a. Set the **Scheduling** option to **Periodic**.
   b. Click **Configure** and specify a period of 30 seconds.
   The calculation will run every 30 seconds.
10. Verify that the results look reasonable:
   a. In the analyses table, right-click the analysis and then click **Preview Results**.
   b. Enter the time range of the preview. The default time range starts at the preceding midnight and ends at the current time, but you can enter different start and end times, if you prefer.
   c. Click **Generate Results**.
11. When you are satisfied with your new analysis, click **Check In** on the toolbar to check in your work.

   PI System Explorer saves and checks in your analysis, and then starts running the analysis.

   A green circle in the **Status** column of the analyses table indicates that PI Analysis Service is running the analysis.
12. Optional. Back fill the PI point with results from the calculation using past data.

   PI Analysis Service can back fill calculation results that have output attributes mapped to PI point data references. However, the PI point cannot contain data at the specified times. If desired, you can delete pre-existing data and then back fill that time range.

   a. Right-click the analysis in the analyses table, and click **Backfill/Recalculate**.
   b. Specify a start and end time for back filling.
   c. Select **Permanently delete existing data and recalculate**.
   d. Click **Start**.

Create a template for power rollup analyses

This example demonstrates how to create a template that you can use to generate a set of rollup analyses. The analysis template aggregates the total power draw of all processes at a refinery.

Before you start

Create the necessary objects in your PI AF database:

- Element template named `Refinery`
- Element named `Rotterdam Refinery`, derived from the `Refinery` template
- Child element named `Catalytic Cracking` under the `Rotterdam Refinery` element
- Attribute named `Power Draw` under the `Catalytic Cracking` child element

Procedure

1. In the Library browser, select the element template for which you want to create the analysis template.

In this example, the element template is called `Refinery`. This template is used to create refinery elements.

2. Click the **Analysis Templates** tab.
3. Create a new analysis template for the element template:
   - If no analysis templates exist, click **Create a new analysis template** to create the first one.
   - Otherwise, click **New Analysis Template** to create a new one.
4. Enter information to identify the analysis:
   - **Name**

     The name that uniquely identifies this analysis. For example, `PowerRollup`.
   - **Analysis Type**

     Click **Rollup**. A rollup analysis calculates statistical data for specified attributes, such as a total or an average value. For more information, see Rollup analyses.
5. Select an example element to provide candidate attributes:
   a. Click the **Select an example element** link to open a window that shows elements derived from the selected element template.
   b. Select an element and click **OK**.

      For example, in the list of elements derived from the `Refinery` template, click **Rotterdam Refinery**.
6. For the **Rollup attributes from** option, click **Child elements of Rotterdam Refinery** to indicate that the analysis rolls up attributes of child elements of the refinery element.
7. Select the attributes to include in the rollup:
   a. From the **Sample Child Element** list, select the element from which you want to view attributes.

      For example, select **Catalytic Cracking** to see attributes from the `Catalytic Cracking` element in the attributes table.

      The table only shows attributes from the selected element. The rollup include attributes from all child elements that match the criteria that you specify.
   b. To the left of the attributes table, specify criteria that select attributes to roll up.

      For example, in the **Attribute Name** field, type `Power*` to specify all attributes that begin with `Power`. The attributes table shows a check mark next to any attributes in the selected child element that match this criteria. However, the rollup includes attributes from any child element that match the criteria, not just those in the table.
8. In the functions table, specify the rollup calculations and output:
   a. Select the check boxes next to any statistical function you want the rollup to calculate.

      For example, click the **Sum** check box to total values.
   b. In the **Output(s)** column, click **Map** to store results from the calculation to an output attribute.

c. Click **New Attribute Template** to create a new attribute template with a PI point data reference.

d. Specify values that define the created attribute and point, and then click **OK**.

For example, you can use the default values, which name the point and attribute `PowerRollup_Sum` and write the point to the default PI Data Archive, indicated by the substitution parameter.

9. Click **Evaluate** and check whether the result returned in the **Value** column is reasonable.
10. Set the **Scheduling** option to **Event-Triggered**.
11. To verify the analysis, examine the results produced with historical data:

    a. In the analyses list, right-click the analysis and then click **Preview Results**.

    b. Enter a start time and end time not later than the current time, and click **Generate Results** to see a list of results.

12. Click **Check In** on the toolbar to save the analysis template to the PI AF server and to create the analysis for any element based on the element template.

Create event frames automatically to track inefficiency

We created a sample analysis to track how the efficiency of a process deviates from the target efficiency in Build an expression analysis to study efficiency deviation. In this example, we create event frames to capture data any time the 15-minute running-average efficiency drops below 90%.

Before you start

Complete the steps in Build an expression analysis to study efficiency deviation.

Procedure

1. In the Library browser, expand **Templates**.
2. Right-click **Event Frame Templates** and then click **New Template** to create a new event frame template.
3. In the **Name** field, enter `EfficiencyAnomaly`.
4. In the **Naming Pattern** field, specify the name of the event frames produced from the template.

   You can use substitution parameters to specify a varying name. PI AF resolves the parameters when creating the event frames. Then, each event frame will have a unique, identifiable name.

   a. Click the arrow next to the field to see valid substitution parameters that you can add to the name.

   b. Select **%ELEMENT%**.

   c. Select **%STARTTIME:yyyy-MM-dd HH:mm:ss.fff%**.

   The **`STARTTIME`** substitution parameter includes the date-time formatting pattern to use.

5. Click the **Attribute Templates** tab.
6. Create a new PI point data reference attribute using an attribute template.
   a. Click **New Attribute Template**.
   b. In the **Name** field, enter `Efficiency`.
   c. Click **Settings** and enter the following configuration string in the **Tag name** field:

      `.\Elements[.]|Efficiency; uom=%`

   d. From the **Default UOM** list, select **percent**.

7. In the Elements pane, click **Ethylene**, the element for which you built an analysis in Build an expression analysis to study efficiency deviation.
8. Click the **Analyses** tab.
9. Click **New Analysis** .
10. Enter information to identify the analysis:
    - **Name**

      Enter `EfficiencyAnomalyEvent`.

- **Analysis Type**

  Click **Event Frame Generation**.

11. From the **Event Frame Template** list, select **EfficiencyAnomaly**.

    This is the event frame template you created in the earlier steps.

12. Enter expressions that specify the start and end of the event frame:

    - **StartTrigger1**

      ```
      TagAvg('Efficiency', '*-15m', '*') <90
      ```

      Creates an event frame whenever the 15-minute average efficiency drops below 90%.

    - **EndTrigger**

      ```
      TagAvg('Efficiency', '*-15m', '*') >92
      ```

      Ends the event frame whenever the 15-minute average efficiency has recovered and exceeds 92%.

13. Click **Evaluate** to check the current values of the expressions.

    The **Value** column is populated with either `True` or `False` for each condition.

14. Specify scheduling for the analysis.

    a. For the **Scheduling** option, click **Periodic**.

    b. Click **Configure** and set the analysis to run every minute.

15. Check in your work.

Backfilling and recalculation of analyses

Backfill or recalculate

When you backfill an analysis for a specified time range, data that are missing are calculated to fill in the gaps for the outputs. Recalculation, on the other hand, deletes all of the data in the time range and calculate results for the analysis outputs again. Currently, there are four different options of backfill and recalculation. For more information, see OSIsoft Tech Support Article KB01697 - Asset analytics backfilling and recalculation milestones and requirements (https://techsupport.osisoft.com/Troubleshooting/KB/KB01697/).

Expression, rollup or SQC analyses can be backfilled or recalculated over an earlier time period if the analyses outputs are mapped to PI point attributes. You can backfill the missing data in a time range and retain existing data, or you can recalculate, which deletes and recalculates data in the time range. Event frame generation analyses can recalculate event frames over a specified time period, but automatically delete all existing event frames in that time period, as well as annotations on affected event frames.

In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. For more information on identity mapping, see PI AF identities and mappings.

> **Note:**
> For analyses writing outputs to attributes configured as PI point, backfilling or recalculation may result in writing out-of-order events. Such events are written to the PI Data Archive without compression.

Enabling automatic backfilling is useful as it fills the gaps in case a short analysis service downtime occurs. For more information, see parameters **`AutoBackfillingEnabled`** and **`MaximumAllowedAutoBackfillingSpanInHours`** in Analysis service configuration.

From PI AF 2017 R2, you can enable automatic recalculation of analyses. PI AF Server and PI Analysis Service 2017 R2 or later are required to make use of automatic recalculation. Although automatic recalculation is globally enabled by default (**`AutoRecalculationEnabled`**), you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation. For more information, see Configure automatic recalculation of analyses and analyses templates, Enable or disable automatic recalculation for multiple analyses and parameters **`AutoRecalculationEnabled`**, **`AutoRecalculationIgnoreTimeInSeconds`**, **`AutoRecalculationMinWaitTimeInSeconds`**, and **`MaxAllowedAutoRecalculationSpanInDays`** in Analysis service configuration. To examine a log file with a list of queued automatic and manual backfill/recalculation requests for troubleshooting, see Open recalculation log folder.

Note:
Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Backfill or recalculate data for an analysis

Before you start

PI Analysis Service 2016 R2 introduced recalculation capabilities for expression, rollup and SQC analyses. PI Data Archive storing the output PI points must be running version 2016 R2 or later. Beginning with PI Analysis Service 2018, you can opt for recalculation of all dependent analyses. An analysis whose input comes from an output of another analysis is considered dependent.

Procedure

1. In the Elements browser, select an element, and click the **Analyses** tab to see the analyses defined for the element.
2. From the list of analyses, right-click the running analysis that you want to use to backfill or recalculate data, and click **Backfill/Recalculate**.
3. In the **Backfilling or recalculation for *analysis name*** window, perform the following actions.
   a. In the **Start Time** and **End Time** fields, specify the time period for which you want to backfill missing data while retaining existing data, or delete and recalculate data. You can use standard PI time expressions or click to select a specific date period.
   b. Decide how existing data should be treated. For an expression, rollup or SQC analysis that outputs to an attribute:
      - To ensure that existing data is retained and only missing data is backfilled, select **Leave existing data and fill in gaps**.
      - If input data or analysis algorithms have changed and you want to delete existing output data, select **Permanently delete existing data and recalculate**.
        - Optional. Check the box next to **Recalculate dependent analyses** to queue all dependent analyses for manual recalculation.

Note:

- Selecting this option may cause recalculation to occur multiple times for dependent analyses
- The recalculation time range for dependent analyses may be different than the time range you selected when queuing recalculation. For more information, see note at the end of step 7 in Backfill or recalculate data for multiple analyses.

Note:
Event frames are automatically deleted and recalculated for event frame generation analyses. Be aware that information such as annotations, acknowledgments, reason codes on those event frames will be lost.

4. Click **Start** to start the backfill or recalculation.
5. For details on the backfill or recalculation status or to cancel, right-click the analysis and click **Backfill/Recalculate Status**.

Video

For information on how to backfill or recalculate data for an analysis, watch this video:

https://www.youtube.com/watch?v=m7F0FwEykpw&rel=0

Backfill or recalculate data for multiple analyses

Before you start

PI Analysis Service 2016 R2 introduced recalculation capabilities for expression, rollup and SQC analyses. PI Data Archive storing the output PI points must be running version 2016 R2 or later. Beginning with PI Analysis Service 2018, you can opt for recalculation of all dependent analyses. An analysis whose input comes from an output of another analysis is considered dependent.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Optional. From **Analysis Searches**, select a search query and then select the result you want to operate upon. For example, select **Enabled**. Only the analyses that meet the criterion ( ) are displayed.
4. In the Analyses pane, select the analyses you want to backfill or recalculate.
5. In the Operations pane, click **Backfill/Recalculate # selected analyses**.

   Additional selection criteria are displayed.
6. In the **Start Time** and **End Time** fields, specify the time period for which you want to backfill missing data while retaining existing data, or delete and recalculate data. You can use standard PI time expressions, or click to select a specific date period.
7. Choose from the following actions:

To backfill or recalculate ...	Do this ...
Expression, rollup, or SQC analyses only	Decide how existing data should be treated: ◦ To ensure that existing data is retained and only missing data is backfilled, select **Leave existing data and fill in gaps**. ◦ If input data or analysis algorithms have changed and you want to delete existing output data, select **Permanently delete existing data and recalculate**.
A mix of analyses that includes event frame generation analyses or More than a page of analyses selections	Perform one of the following actions: ◦ Keep **Leave existing data and fill in gaps** selected and acknowledge that event frames in the time range will be permanently deleted along with all annotations on those event frames. ◦ Select **Permanently delete existing data and recalculate**. All annotations on event frames will be lost. ▪ Optional. Check the box next to **Recalculate dependent analyses** to queue all dependent analyses for manual recalculation. **Note:** ◦ Selecting this option may cause recalculation to occur multiple times for dependent analyses ◦ The recalculation time range for dependent analyses may be different than the time range you selected when queuing recalculation

Note:

Three factors affecting the recalculation time range of dependent analyses

- Time range of the original analysis
- How the output of the original analysis is used in the dependent analyses
- Whether or not the output time override is configured in the original analysis

8. Click **Queue** to start multiple backfills or recalculations.
9. Review the **Pending Operations** pane for details on the backfill or recalculation status.

Reconciliation of event frames during backfilling

When PI Analysis Service is restarted after short down-times, it initiates automatic backfilling concurrently with real-time analysis.

> **Note:**
> With PI Server 2015 and PI Server 2015 R2, PI Analysis Service does not initiate the automatic backfilling process if down-time is greater than 72 hours. With PI Server 2016, you can configure the **`MaximumAllowedAutoBackfillingSpanInHours`** configuration parameter in the PI Analysis Service to specify the desired time.

With PI Server 2015 R2, when PI Analysis Service restarts, it closes event frames that were open, indicating this reason in the event frame description field. When automatic backfilling for analyses is completed, Asset Analytics determines if there are event frames to be reconciled and, accordingly, starts the reconciliation phase. Event frame reconciliation may be necessary for any in-progress event frames that cannot be closed within the auto-backfilling time range because no closing event can be found. These event frames may be generated in either of these ways:

- The event frame may be already present when the analysis service was stopped
- The event frame may be newly created during auto-backfilling

Such in-progress event frames may have to be reconciled with those generated during real-time evaluation.

When you select an analysis using the Analysis tab or the Analysis Management plug-in, an orange icon ( ) in the analysis window indicates that the event frame reconciliation is pending completion. Hover over the icon to see details such as the number of events being processed, start and end times, and the time of last evaluation. When reconciliation is complete, the icon turns from orange to green.

Configure automatic recalculation of analyses and analyses templates

Before you start

From PI AF 2017 R2, you can automatically recalculate analyses. Although automatic recalculation is globally enabled by default, you need to opt in at individual analysis (template) level if you expect late-arriving or out-of-order data that may trigger recalculation. To configure automatic recalculation for a particular analysis (template), follow this procedure.

> **Note:**
> - PI AF Server and PI Analysis Service 2017 R2 or later are required.
> - Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Procedure

1. Go to the analysis (template) for which you want to enable automatic recalculation.
2. Click **Advanced** button at the bottom of the pane. Advanced options window opens.
3. Check the box next to **Recalculate analysis for out-of-order input events** and click **OK**.

   > **Note:**
   > You cannot opt in or out of automatic recalculation by accessing the analysis if it is based on a template. Go to the template to make changes.

Enable or disable automatic recalculation for multiple analyses

Before you start

For analyses writing to attributes (expression, rollup and SQC), automatic recalculation is available for PI Analysis Service 2017 R2 or later. In addition, PI AF Server 2017 R2 or later is required.

> **Note:**
> Event frame generation and SQC analyses that are configured to generate event frames are excluded from automatic recalculation.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Optional. From **Analysis Searches**, select a search query and then select the result you want to operate upon. For example, select **Enabled**. Only the analyses that meet the criterion ( ) are displayed.
4. In the Analyses pane, select the analyses you want to enable or disable automatic recalculation.
5. In the Operations pane, click either **Enable automatic recalculation for selected analyses** or **Disable automatic recalculation for selected analyses**.
6. Check the box next to acknowledgement and click **Enable** or **Disable**.
7. Review the **Pending Operations** pane to track the progress of enabled or disabled operation.

Open recalculation log folder

To access a comprehensive log file of queued automatic/manual backfilling and recalculation requests for troubleshooting, follow the procedure.

Procedure

1. In the navigator, select **Management** at the bottom of the list.
2. Select **Analyses** radio button in the **Management** pane.
3. Right-click anywhere in the **Operations** panel and click **Open Recalculation Log folder**. A folder containing a csv file with queued backfill/recalculation opens.

Management of analyses in a PI AF database

The Management feature (which includes analyses and notification rules) is available if your installation of PI System Explorer includes the Management plug-in.

For more information, go to Management of analyses. To search for analyses in your database, see Search for analyses or notification rules. For information on notifications management, see Management of notification rules.

Expression functions reference

Expression functions available for use in expression and event frame generation analyses are listed alphabetically. These functions follow performance equation (PE) syntax.

Topics in this section

- Abs
  Return the absolute value of an integer or real number.
- Acos
  Return the inverse cosine (arccos) of an integer or real number.
- AND
  Operator for the logical conjunction of two expressions.
- ArrayLength
  Get the length of an array.
- Ascii
  Return the ASCII character code of the first character of a string.
- Asin
  Return the inverse sine (arcsin) of a number.
- Atn
  Return the inverse tangent (arctan) of a number.
- Atn2
  Return the arctan for a specified tangent value a/b.
- Avg
  Return the average of one or more values.
- BadVal
  Test a value to see if it is bad.
- Bod
  Return a timestamp for beginning of the day from a time expression.
- Bom
  Return a timestamp for midnight on the first day of the month from a given time expression.
- Bonm
  Return a timestamp for midnight on the first day of a following month from a given time expression.
- Ceiling
  Return the smallest integer not less than x.
- Char
  Return a string built from specified ASCII character codes.
- Compare
  Compare two strings using wildcard characters ("*" and "?").
- Concat
  Concatenate two or more strings.
- Contains
  Determine if first string contains second string and return `True` or `False`.

- Convert
  Convert a value from its current unit of measure (UOM) to a specified UOM.
- Cos
  Return the trigonometric cosine of an integer or real number.
- Cosh
  Return the hyperbolic cosine of a number.
- Cot
  Return the trigonometric cotangent of a number
- Coth
  Return the hyperbolic cotangent of a number.
- Cov
  Determine the covariance between two sets of values given by attributes over a specified time range.
- Csc
  Return the trigonometric cosecant of a number.
- Csch
  Return the hyperbolic cosecant of a number.
- Curve
  For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$.
- Day
  Extract the day of the month from a time expression.
- DaySec
  Return the total time in seconds between the start of day (midnight) and the time denoted in the argument.
- DeltaValue
  Return the difference between the current and immediately previous values or evaluations of an attribute with a numeric or DateTime value type.
- DigState
  Translate a character string representing a digital state into its corresponding digital state.
- DigText
  Obtain the text corresponding to the digital state.
- E
  Constant function that returns the value for **e** (the base of the natural logarithm).
- ELSE
  Operator that returns the second of two specified values when the conditional expression in IF-THEN-ELSE statement is True.
- EventCount
  Find the number of good events for an attribute during a specified time range.
- EventFrame
  Return the value of event frame properties.
- Exp
  Return the exponential of an integer or real number.

- FilterData
  Get all the values in an array that satisfy a given condition.
- FindEq
  Find the first time within a time interval when an attribute is equal to a specified value.
- FindGE
  Find the first time within a time interval when an attribute is greater than or equal to a specified value.
- FindGT
  Find the first time within a time interval when an attribute is greater than a specified value.
- FindLE
  Find the first time within a time interval when an attribute is less than or equal to a specified value.
- FindLT
  Find the first time within a time interval when an attribute is less than a specified value.
- FindNE
  Find the first time within a time interval when an attribute is not equal to a specified value.
- FirstValue
  Get the first value in an array that satisfies a given condition.
- Float
  Convert a string to a number.
- Floor
  Return the largest integer not greater than *x*.
- Format
  Convert a number to string according to a format expression.
- Frac
  Return the fractional part of a real number. Returns 0 for integers.
- HasChanged
  Return `True` if an attribute has had any event in the specified time period; otherwise return `False`.
- HasValueChanged
  Determine if the value of an attribute or expression has changed since last evaluated during an analysis.
- Hour
  Extract the hour from a time expression.
- IF
  The condition for an IF-THEN-ELSE statement.
- IN
  Operator that indicates whether a value is included in a set or range of values.
- InStr
  Determine the location within a string where a sub-string match is first found.
- Int
  Return the integer part of an integer or real number.

- InterpolatedValues
  Obtain interpolated values over a specified time range using the specified time step.
- IsSet
  For an attribute associated with a PI point, determine if the point is annotated, substituted, or questionable.
- LastValue
  Get the last value in an array that satisfies a given condition.
- LCase
  Convert a string to a lowercase string.
- Left
  Determine a specified number of characters of a string from the left.
- Len
  Determine the length of a string.
- LinRegr
  Return a one-based array of slope, intercept and R^2 based on time stamped values.
- Log
  Return the natural logarithm (base e = 2.7182818...) of an integer or real number.
- Log10
  Return the base 10 logarithm of an integer or real number.
- Logbase
  The logarithm of a positive numeric value to a specified base.
- LTrim
  Remove the leading blanks from a string.
- MapData
  Apply a function to each value in an array.
- Max
  Return the maximum of a set of values.
- Median
  Return the median (middle) value for a set of one or more values.
- Mid
  Return a sub-string within a string.
- Min
  Return the minimum of a set of values.
- Minute
  Extract the minute from a time expression.
- Mod
  Operator that returns the remainder from the quotient of two numeric values.
- Month
  Extract the month from a time expression.
- NextEvent
  Find the timestamp of the next recorded value after a specified time.
- NextVal
  Find the next recorded value after a specified time.

- Noon
  Return a timestamp for noon on the day of a given time expression.
- NoOutput
  Do not write current calculation result.
- Normalrnd
  Return a random number that maps the normal distribution curve using a specified mean and standard deviation.
- NOT
  Operator that returns the negation of an expression.
- NumOfChanges
  Return the number of changes in value for an attribute within a specified time range.
- OR
  Operator for the logical disjunction of two expressions.
- ParseTime
  Translate a PI time expression to a timestamp.
- PctGood
  Find the time percentage, over a specified time range, for which the attribute had good values.
- Pi
  Constant function that returns the value for pi.
- Poisson
  Return a random number that maps a Poisson distribution with a specific mean.
- Poly
  Return the polynomial $c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$.
- Pow
  Return **x** raised to the power **y**.
- PrevEvent
  Find the timestamp of the last recorded value before a specified time.
- PrevVal
  Find the last recorded value before a specified time.
- PStDev
  Return the *population standard deviation* for a population of one or more values.
- Rand
  Returns a random number from a specified range.
- Range
  Find the difference between the maximum and minimum values for an attribute during a specified time range.
- Rate
  Return the difference over time between the current value and a previously evaluated value of an attribute or (time) expression.
- RecordedValues
  Obtain values within a particular time range based on user-specified boundary type.

- RecordedValuesByCount
  Obtain a specified number of values beginning at the requested start time in the direction specified.
- Remainder
  Return the remainder resulting from the division of x by y using the IEEERemainder method.
- Right
  Determine a specified number of characters of a string from the right.
- Round
  Round a number or time to the nearest unit.
- Roundfrac
  Round a value to a specified number of decimal places.
- RTrim
  Trim trailing blanks from a string.
- Sec
  Return the trigonometric secant of a number.
- Sech
  Return the hyperbolic secant of a number.
- Second
  Extract the second from a time expression.
- SecSinceChange
  Return the time in seconds since an attribute value was updated.
- Sgn
  Return an indicator of the numerical sign of a number.
- Sin
  Return the trigonometric sine of a number.
- Sinh
  Return the hyperbolic sine of a number.
- Sqr
  Return the square root of a number.
- StateNo
  Translate a digital state into its corresponding enumeration value.
- SStDev
  Return the *sample standard deviation* for two or more arguments that are a sample of a larger population.
- StDev
  Find the time-weighted standard deviation of values for an attribute from a specified time range.
- String
  Convert any value to a string.
- TagAvg
  Find the time-weighted average of values for an attribute during a specified time range.

- TagBad
  Test the point for an abnormal state at a specified time
- TagDesc
  For an attribute associated with a PI point, retrieve that point's descriptor from the point database.
- TagEU
  For an attribute associated with a PI point, retrieve the point's engineering unit string from the point database.
- TagExDesc
  For an attribute associated with a PI point, retrieve the point's extended descriptor from the point database.
- TagMax
  Find the maximum value of an attribute during a specified time range.
- TagMean
  Find the average (mean) of values for an attribute during a specified time range.
- TagMin
  Find the minimum value of an attribute during a specified time range.
- TagName
  For an attribute associated with a PI point, get a point's name from the point database.
- TagNum
  For an attribute associated with a PI point, get a point's number from the point database.
- TagSource
  For an attribute associated with a PI point, get a point's point source string from the point database.
- TagSpan
  For an attribute associated with a PI point, get a point's span from the point database.
- TagTot
  Find the totalized value (time integral) of an attribute's values over a specified time range.
- TagType
  For an attribute associated with a PI point, get a point's point type from the point database.
- TagTypVal
  For an attribute associated with a PI point, get a point's typical value from the point database.
- TagVal
  Return the value of an attribute at a specified time.
- TagZero
  For an attribute with a PI point data reference, get a PI point's "zero" value from the point database.
- Tan
  Return the trigonometric tangent of a number.
- Tanh
  Return the hyperbolic tangent of a number.
- Text
  Concatenate strings representing argument values.

- THEN
  Operator that returns the first of two specified values when the conditional expression in IF-THEN-ELSE statement is True.
- TimeEq
  Find the total time, within a range, when an attribute value is equal to a specified value.
- TimeGE
  Find the total time, within a range, when an attribute value is greater than or equal to a specified value.
- TimeGT
  Find the total time, within a range, when an attribute value is greater than a specified value.
- TimeLE
  Find the total time, within a range, when an attribute value is less than or equal to a given value.
- TimeLT
  Find the total time, within a range, when an attribute value is less than a specified value.
- TimeNE
  Find the total time, within a range, when an attribute value is not equal to a specified value.
- TimeStamp
  Return the time stamp for a single time-stamped value or an array of time-stamped values.
- Total
  Return the sum of two or more values.
- Trim
  Trim blanks on both sides of a string.
- Trunc
  Truncate a number or time to the next lower unit.
- UCase
  Convert a string to an uppercase string.
- Weekday
  Extract the day of the week from a timestamp.
- Year
  Extract the year from a time expression.
- Yearday
  Extract the day of the year from a time expression.

Abs

Return the absolute value of an integer or real number.

Syntax

```
Abs(x)
```

Arguments

- **x**

  An integer or real number

Returns

The absolute value of *x*

Exceptions

Return an error value if *x* is not an integer or real number

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Abs(-2.2)`

  [Returns 2.2]

- `Abs('att1')`

  [Return the absolute value of 'att1' at trigger time]

Acos

Return the inverse cosine (arccos) of an integer or real number. The inverse cosine of *x* is the angle in radians whose cosine is equal to *x*.

Syntax

```
Acos(x)
```

Arguments

- **x**

  Must be a real number between -1.0 and 1.0, inclusive

Returns

The inverse cosine of *x*, in radians

Exceptions

If *x* is not a number, or is less than -1.0 or greater than 1.0, returns an error value

Example

- `Acos(-0.5)`

  [Returns 2.0944]

- `Acos(0.75)`

  [Returns 0.72273]

AND

The logical conjunction of two expressions that returns **`True`** if both expressions are true and **`False`** otherwise.

Syntax

```
expression1 AND expression2
```

Arguments

- **`expression1, expression2`**

  Any expression that evaluates to true or false

Returns

`True` if both expressions are true (non-zero) and **`False`** (zero) otherwise

Exceptions

None

Example

- `('att1' > 50) AND ('att2' = "good")`

ArrayLength

Get the length of an array.

Syntax

```
ArrayLength(array)
```

Arguments

- **`array`**

  a variable representing the array to operate on

Returns

The number of values in the array

Exceptions

None

Notes

None

Example

- `ArrayLength(Data)`

  [Return the number of values in the array named **`Data`**]

Ascii

Return the ASCII character code of the first character of a string.

Syntax

```
Ascii(s1)
```

Arguments

- **s1**

  Any expression evaluating to a string

Returns

The character code of the first character of the string

Exceptions

If the argument is not a string, an error value is returned

Example

- `Ascii("D")`

  [Returns 68, the ASCII character code for D]

- `Ascii("Program")`

  [Returns 80, the ASCII character code for the first letter of the string]

Asin

Return the inverse sine (arcsin) of a number. The inverse sine of *x* is the angle in radians whose sine is equal to *x*.

Syntax

```
Asin(x)
```

Arguments

- **x**

  Must be a real number between -1.0 and 1.0, inclusive

Returns

The inverse sine of *x*, in radians

Exceptions

If *x* is not a number, or is less than -1.0 or greater than 1.0, an error value is returned

Example

- `Asin(-0.5)`

  [Returns -0.5236]

- `Asin(TagVal('att1','y+8h'))`

  [Return the inverse sine of the value of 'att1' at 8am yesterday]

- `Asin('att1')`

[Return the inverse sine of the value of 'att1' at trigger time]

Atn

Return the inverse (or arc) tangent of an integer or real number. The inverse tangent of *x* is the angle in radians whose tangent is equal to *x*.

Syntax

```
Atn(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The inverse tangent of *x*, in radians

Exceptions

Returns an error if *x* is not an integer or real number

Example

- `Atn(1)`

  [Returns 0.7854]

- `Atn(-2.2)`

  [Returns -1.1442]

- `Atn('att1')`

  [Return the inverse tangent of the value of 'att1' at trigger time]

- `Atn('att1' - 'att2')`

  [Return the inverse tangent of the difference of 'att1' and 'att2' at trigger time]

Atn2

Return the inverse tangent (arctan) of a tangent value a/b. The inverse tangent is the angle measured in radians from the positive x-axis to a line whose endpoints are the origin and the Cartesian coordinates (b, a).

Syntax

```
Atn2(x, y)
```

Arguments

- **x**

  An integer or real number

- **y**

  A non-zero integer or real number

Returns

The inverse tangent in radians of the tangent value x/y

Exceptions

Returns an error if *x* or *y* is not an integer or real number

Example

- `Atn2(1,2)`

  [Returns 0.46365]

- `Atn2('att1', 'att2')`

  [Return the inverse tangent in radians of the tangent value 'att1'/'att2' at trigger time]

- `Atn2(TagVal('att1','y+8h'), TagVal('att2', 'y+8h'))`

  [Return the inverse tangent in radians of the tangent value 'att1'/'att2' at 8am yesterday]

Avg

Return the average from a set of one or more values.

Syntax

```
Avg (x1 [, ... xn])
```

Arguments

- **x1, ... xn**

  Arguments or a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The average of one or more values. The result is of the same data type as arguments.

Exceptions

Arguments whose run-time values are character strings or digital states are not included in the average. If all values are character strings or digital states, Avg returns an error value. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs. Avg returns not time-based but event-based average.

Example

- `Avg(14, 'att1', 14.5, TagVal('att2', '14-Dec-16'))`

[Find the average of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at the start of day (12:00am) on Dec 14, 2016]

- `Avg('*', 'y+8h', 'Saturday')`

  [Find the average of these time stamps: now, 8:00am yesterday, and start of day (12:00am) last Saturday]

- `Avg('*'-'*-1h', 't+4h'-'y+8h', 'y+2h'-'20')`

  [Find the average of these time periods: from 1 hour ago to now, from 8:00am yesterday to 4:00am today, and 2:00am yesterday to the start of day (12:00am) on the 20th of this month]

- `Avg(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Avg(Variable1)	5

  [Find the average of values in an array `Variable1`]

BadVal

Test a value to see if it is bad. For an attribute associated with a PI point, any system digital state value is considered bad (`No Data` or `Calc Failed`, for example).

Syntax

```
Badval(x)
```

Arguments

- **x**

  any expression evaluating to a value

Returns

`True` if the value is bad

`False` if the value is not bad

Exceptions

Returns `True` for blob PI points. Returns `False` for string PI points

Example

- `BadVal(1/0)`

  [Returns `True`]

- `BadVal(10)`

  [Returns `False`]

- `BadVal('att1')`

  [Returns `True` if the value of 'att1' is bad]

- `BadVal(FindEq('att1', 't', '*', 10))`

  [Returns `True` if the embedded function (**`FindEq`** in this example) has no result or has error for any reason]

Bod

Return a timestamp for beginning of the day from a time expression.

Syntax

```
Bod(t1)
```

Arguments

- **`t1`**

  A time expression, enclosed in single quotes

Returns

Timestamp for the start of the day

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bod('*')`

  [Return a timestamp for beginning of the day today.]

- `Bod('y')`

  [Return a timestamp for beginning of the day yesterday.]

- `Bod(FindEq('att1', '-3d', '*', 50))`

  [Return a timestamp for beginning of the day when 'att1' value was first equal to 50 in the past 72 hours.]

Bom

Return a timestamp for midnight on the first day of the month from a given time expression.

Syntax

```
Bom(t1)
```

Arguments

- **t1**

  A time expression

Returns

Timestamp for the start of the month.

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bom('*')`

  [Return a timestamp for midnight on the first day of this month.]

- `Bom(PrevEvent('att1', '*'))`

  [Return a timestamp for midnight on the first day of the month when 'att1' had a value before the current one.]

- `Bom(FindEq('att1', '-60d', '*', 50))`

  [Return a timestamp for midnight on the first day of the month when the value of 'att1' was first equal to 50 in the past 60 days.]

Bonm

Return a timestamp for midnight on the first day of a following month from a given time expression.

Syntax

```
Bonm(t1)
```

Arguments

- **t1**

  Time expression, enclosed in single quotes

Returns

Timestamp for the start of the next month.

Exceptions

None

Notes

This function is useful for establishing a time at a unique clock time independent of the length of particular days.

Example

- `Bonm('*')`

  [Return a timestamp for midnight on the first day of next month.]

- `Bonm('y')`

  [Return a timestamp for midnight on the first day of the following month from yesterday's date.]

- `Bonm(FindEq('att1', '-60d', '*', 50))`

  [Return a timestamp for midnight on the first day of the following month when the value of 'att1' was first equal to 50 in the past 60 days.]

Ceiling

Return the nearest integer greater than or equal to *x*.

Syntax

```
Ceiling(x)
```

Arguments

- **x**

  integer or a real number, an attribute whose value evaluates to a number

Returns

The nearest integer greater than or equal to *x*

Exceptions

Returns an error if *x* is a digital state, time expression, time interval or a string

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Ceiling(2.3)`

  [Returns 3]

- `Ceiling(-2.3)`

  [Returns -2]

- `Ceiling(TagVal('att1', '12/30/16'))`

  [Return the nearest integer to the value of 'att1' at the start of day (12:00am) on December 30, 2016]

Char

Build a string from ASCII character codes.

Syntax

```
Char(x1, ... xn)
```

Arguments

- **`x1, ... xn`**

  Integers

Returns

A string built from the 80 character codes

Exceptions

Returns an error if an argument is not a number

Example

- `Char(80, 73)`

  [Returns "PI"]

- `Char(65)`

  [Returns "A"]

- `Char(5 * 10)`

  [Returns "2"]

Compare

Compare two strings using wildcard characters ("*" and "?") or compare two operands using logical operators and deadband value.

Syntax

```
Compare {[(s1, s2 [, caseSensitive])]|[x1, x2, op, deadband]}
```

Arguments

- **`s1, s2`**

  string (s2 can contain wildcard characters)

- **`caseSensitive`**

  Optional flag indicating if the comparison is case sensitive. If `False` (the default), the comparison is not case-sensitive. If `True`, the comparison is case-sensitive.

- **`x1, x2`**

  Numeric value (integer or real number) or time expression

- **op**

  An operator used for comparison. Must be one of the following operators, or a variable defined as one of these: <, >, <=, >=

- **deadband**

  Numeric value (integer or real number) or timespan. *Deadband* specifies a buffer (tolerance) around *x2* and prevents repeated alerts when the value fluctuates within the *deadband* range.

Returns

- `True` if *s1* = *s2*
- `True` if *x1* <op> *x2* is true, and uses *deadband* value for subsequent evaluations

`False` otherwise

Exceptions

Wildcard characters in *s1* are treated literally and not as wildcards.

The deadband value is applied to *x2*, not *x1*.

Example

- `Compare('att1', 100, "<=", 5)`

  [Returns `True` if 'att1' is less than or equal to 100, and uses the deadband value of 5 for subsequent evaluations]

- `Compare("What", "what", True)`

  [Returns `False`]

- `Compare("b", "a")`

  [Returns `False`]

- `Compare("What", "wha?")`

  [Returns `True`]

- `Compare("What", "wh")`

  [Returns `False`]

Concat

Concatenate two or more strings.

Syntax

```
Concat(s1, s2 [, ... sn])
```

Arguments

- **s1, ... sn**

  Must be character strings, or expressions yielding character strings.

Returns

The character strings, concatenated together. This function does not insert blanks between its arguments. To include a space in the concatenated string, add an argument consisting of a string that has a single space enclosed in double quotes.

Example

- `Concat("shut", "down")`

  [Returns "shutdown"]

- `Concat("shut ", "down")`

  [Returns "shut down"]

Contains

Determine if first string contains second string and return `True` or `False`.

Syntax

```
Contains(s1, s2 [,caseSensitive])
```

Arguments

- **`s1`**

  Primary string

- **`s2`**

  Substring searched for in primary string

- **`caseSensitive`**

  Optional: Boolean that indicates whether the search is case sensitive. If `False` (or omitted), the search is not case sensitive. If `True`, the search is case sensitive

Returns

`True` if the primary string (*s1*) contains the substring (*s2*) and `False` otherwise

Exceptions

If either of the first two arguments (*s1* or *s2*) is not a string, the function returns an error value

If the optional third argument (*caseSensitive*) is not a Boolean, the function returns an error value

Example

- `Contains("What","Hat",True)`

  [Returns `False`]

- `Contains("What","Hat")`

  [Returns `True`]

- `Contains("What","at")`

[Returns `True`]

- `Contains("hat","what")`

  [Returns `False`]

Convert

Convert a value from its current unit of measure (UOM) to a specified UOM; for a value with no UOM, assign the specified UOM.

Syntax

```
Convert(x, toUnit)
```

Arguments

- **`x`**

  Any expression that resolves to a numeric value, including the name of a numeric attribute enclosed in single quotation marks, a variable name, or a constant value

- **`toUnit`**

  The output UOM enclosed in double quotation marks

Returns

The numeric value converted to the specified UOM

Exceptions

Returns an error if *toUnit* and the initial UOM for *x* are not in the same UOM class

Example

- `Convert('MetricWeight', "lb")`
- `Convert('Ambient Temperature', "degC")`

  Note:
  PI AF substitutes `deg` with `°`, if not otherwise found in the lookup.

Cos

Return the trigonometric cosine of an integer or real number.

Syntax

```
Cos(x)
```

Arguments

- **`x`**

  Must be an integer or real number, which represents an angle in radians

Returns

The cosine of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Cos(1.1)`

  [Returns 0.4536]

- `Cos(1)`

  [Returns 0.5403]

- `Cos('att1')`

  [Return the trigonometric cosine of the value of 'att1' at trigger time]

Cosh

Return the hyperbolic cosine of a number.

Syntax

```
Cosh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cosine of *x*

Exceptions

If *x* is not an integer or real number, returns an error

Example

- `Cosh(1)`

  [Returns 1.5431]

- `Cosh(-1)`

  [Returns 1.5431]

- `Cosh('att1')`

  [Return the hyperbolic cosine of the value of 'att1' at trigger time]

Cot

Return the trigonometric cotangent of a number.

Syntax

```
Cot(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The cotangent of *x*

Exceptions

If *x* is not a number, returns an error

Example

- `Cot(1)`

  [Returns 0.64209]

- `Cot(1.1)`

  [Returns 0.50897]

- `Cot('att1')`

  [Return the trigonometric cotangent of the value of 'att1' at trigger time]

Coth

Return the hyperbolic cotangent of a number.

Syntax

```
Coth(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cotangent of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Coth(1)`

[Returns 1.313]

- `Coth(1.1)`

  [Returns 1.2492]

- `Coth('att1')`

  [Return the hyperbolic cotangent of the value of 'att1' at trigger time]

Cov

Determine the covariance between two sets of values given by attributes over a specified time range.

Syntax

```
Cov(x, y, mode, starttime, endtime, type [, pctgood])
```

Arguments

- **`x`**

  an attribute with the first set of time series data (such as PI point data reference) enclosed in single quotes

- **`y`**

  an attribute with the second set of time series data (such as PI point data reference) enclosed in single quotes

- **`mode`**

  a number that specifies how to align time-stamped values

  choose from 0, 1 or 2 where:

  0 represents the combination of time-stamped values from *x* and *y*

  1 represents values from both attributes according to *x*'s time stamps

  2 represents values from both attributes according to *y*'s time stamps

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute *endtime*

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute *starttime*

- **`type`**

  a number specifying the covariance type: use 0 for "population" and 1 for "sample"

- **`pctgood`**

  Optional. Minimum percentage of time during the time range that attribute values must be good.

You can set *pctgood* as a threshold to ensure that there are sufficient good values to calculate **Cov**.

Returns

Covariance of *x* and *y*

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value

Notes

Bad values are excluded from **Cov** calculation

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good

Example

- `Cov('att1', 'att2', 0, 't', '+1h', 1, 80)`

  [Return the sample covariance of 'att1' and 'att2' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `Cov('att1', 'att2', 1, 't', '+1h', 1)`

  [Return the population covariance of 'att1' and 'att2' based on time stamps of 'att1' between 12:00 and 1:00am today]

Csc

Return the trigonometric cosecant of a number.

Syntax

```
Csc(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The cosecant of *x*

Exceptions

If *x* is not a number, returns an error

Example

- `Csc(1)`

[Returns 1.1884]

- `Csc(1.1)`

  [Returns 0.7487]

- `Csc('att1')`

  [Return the trigonometric cosecant of the value of 'att1' at trigger time]

Csch

Return the hyperbolic cosecant of a number.

Syntax

```
Csch(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic cosecant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Csch(1)`

  [Returns 0.85092]

- `Csch(1.1)`

  [Returns 0.748]

- `Csch('att1')`

  [Return the hyperbolic cosecant of the value of 'att1' at trigger time]

Curve

For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$.

Syntax

```
Curve(x, (x1,y1) (x2,y2) … (xn,yn) )
```

Arguments

- **x**

  Expression evaluating to a number

- **x_1, y_1**

  The first point on the curve. The x_i's and y_i's are numeric constants evaluated at compile time. The values set for x_i's must be in ascending order.

Returns

For a given *x*, returns the value of a piecewise linear interpolation function defined by the set of n points $(x_1,y_1)...(x_n,y_n)$. If the value of *x* is less than x_1 then y_1 is returned and if it is greater than x_n, y_n is returned. The points are assumed to be ordered in the *x* direction from smallest to largest.

Exceptions

If the value of *x* is not an integer or real number, an error value is returned

Example

- `Curve(3, (0,100) (100,0) )`

  [Returns 97]

- `Curve('att1', (25,25) (75,75) )`

Day

Extract the day of the month from a time expression.

Syntax

```
Day(t1)
```

Arguments

- **t1**

  time expression enclosed in single quotes

Returns

The day of the month of a time expression in the range 1 through 31.

Exceptions

None

Example

- `Day('*')`

  [Return what day of the month today is.]

- `Day('y')`

  [Return what day of the month yesterday was.]

- `Day(FindEq('att1', '-28d', '*', 50))`

  [Return what day of the month it was when the value of 'att1' was first equal to 50 in the past 28 days .]

DaySec

Return the total time in seconds between the start of day (midnight) and the time denoted in the argument.

Syntax

```
DaySec(t1)
```

Arguments

- **t_1**

  A time expression, enclosed in single quotes

Returns

Total seconds since the start of day (midnight) till *t1*, in the range 0-86399

Exceptions

None

Example

- `DaySec('*')`

  [Return the number of seconds from the start of day (midnight) until now]

DeltaValue

Return the difference between the current and immediately previous values or *evaluations* of an attribute with a numeric or DateTime value type.

Syntax

```
DeltaValue(x [, prevEval])
```

Arguments

- **x**

  An attribute or expression of a numeric or DateTime value type

- **prevEval**

  (Boolean) If `False` or not specified, *x* must be an attribute, and **`DeltaValue`** returns the difference between its current value and immediately previous value.

  If `True`, **`DeltaValue`**returns the difference between the current and immediately previous *evaluations* of *x*, which can be any expression with a numeric or DateTime value type

Returns

Returns the difference between the current value and the immediately previous value of an attribute with a numeric or DateTime value type, or the difference between the current and immediately previous *evaluations* for any expression with a numeric or DateTime value type

Notes

When *x* is of DateTime data type, **`DeltaValue`** returns the time interval (in seconds) between two DateTime values. If the time interval value is output to an attribute, make sure the attribute data type is set to double

This function may use input values from PI points. If the compression is on for a PI point, all of its values may not be preserved in PI Data Archive. Before you try to validate the results of this function using a client tool such as PI Datalink, make sure the compression is turned off. This is to ensure that all associated PI point values are included for validation. When validation is complete, be sure to turn the compression back on again as a best practice

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `DeltaValue('att1' [, FALSE])`

  [Return the difference between the current and immediately preceding value of 'att1']

- `DeltaValue('att1', TRUE)`

  [Return the difference between the current and last *evaluated* value of 'att1']

DigState

Convert a string into a corresponding PI Data Archive digital state object, either based on the attribute's digital state enumeration or on a system enumeration set values (for example, `Calc Failed`). Use **`DigState`** embedded in other functions when PI Data Archive digital state object is required for input (such as with **`StateNo`**) or in conjunction with other functions for comparison with digital state attributes.

Syntax

```
DigState(s1 [, x])
```

Arguments

- **s_1**

  A string representing a digital state in double quotes

- **x**

  Optional: An attribute with PI point digital state reference or of value type enumeration set. If omitted, only the system digital set is searched for the given string.

Returns

An enumeration value

Exceptions

If the string does not represent a digital state of the specified attribute, the function returns `Calc Failed`. If the attribute is omitted and string does not represent a system digital state, `Calc Failed` is returned.

Example

- `'att1' > DigState("Program", 'att1')`

  [Returns `True` if current digital state of 'att1' is greater than the digital state represented by "Program" value]

- `DigState("No Result")`

  [Construct a value with system digital state `No Result`]

DigText

Obtain the text corresponding to the digital state.

Syntax

```
DigText(digstate)
```

Arguments

- **`digstate`**

  An argument that evaluates to a digital state

Returns

The text for the digital state

Exceptions

Returns an error if the argument is not a digital state

Example

- `DigText(TagVal('enum_att1', 'y'))`

  [Returns the digital state value in string]

- `DigText('enum_att1')`

  [Returns digital state value in string. Returns an error message if 'enum_att1' is not a PI point attribute with digital state value]

E

Return the value for **e** (the base of the natural logarithm).

Syntax

```
E()
```

Arguments

None

Returns

The value for **e** (the base of the natural logarithm).

Exceptions

None

Example

- `E()`

ELSE

Operator that returns the second of two specified values when the conditional expression in IF-THEN-ELSE statement is True.

Syntax

```
IF (expression) THEN (x) ELSE (y)
```

Arguments

- **`expression`**

  Any expression that evaluates to true or false

- **`x, y`**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return the value of 'att2'; otherwise return the value of 'att3']

EventCount

Find the number of good events for an attribute during a specified time range.

Syntax

```
EventCount(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of good values for events within the specified time interval.

Returns

Number of events for the attribute during a specified time range.

Exceptions

If there are no good values or the **pctgood** minimum is not reached for the given time interval, an error is returned.

Notes

Bad values are excluded from **EventCount** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `EventCount('att1', 't', '+1h')`

  [Return the count of events between 12:00 and 1:00am today.]

- `EventCount('att1', 't', '+1h', 80)`

  [Return the count of events between 12:00 and 1:00am today when at least 80% of the values were good.]

EventFrame

Return the value of event frame properties.

Syntax

```
EventFrame(parameter)
```

Arguments

- **parameter**

  parameter string *StartTime*, *EndTime* or *Duration* enclosed in double quotes; not case-sensitive

Returns

The value of the parameter

Exceptions

If the event frame is not active or still open, returns `No Results`

Notes

- This function is only to be used when configuring output expressions in an event frame generation analysis
- The output can be mapped to an attribute with value type DateTime for *StartTime* and *EndTime*. For *Duration*, select double (time returned in seconds)
- You can preview results by right-clicking the analysis

Example

- `EventFrame("Duration")`

  [Return the duration of an event frame. Time is returned in seconds]
- `TagAvg('att1', EventFrame("StartTime"), EventFrame("EndTime"))`

  [Find the time-weighted average of the values of 'att1' during an event frame]

Exp

Return the exponential of an integer or real number. This is the number e^x, where e = 2.7182818...

Syntax

```
Exp(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The exponential of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Exp(11)`

  [Returns 59874]
- `Exp('att1')`

  [Return the exponential of the value of 'att1' at trigger time]
- `Exp(TagVal('att1','t'))`

  [Return the exponential of the value of 'att1' at the start of day (12am) today]

FilterData

Get all the values in an array that satisfy a given condition.

Syntax

```
FilterData(array, condition($val))
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

An array of all the values that satisfy the given condition

Exceptions

None

Notes

None

Example

- `FilterData(Data, (NOT BadVal($val) AND $val > 50))`

  [From an array named **`Data`**, return a new array of values where none are bad values and all are greater than 50]

FindEq

Find the first time within a time interval when an attribute is equal to a specified value.

Syntax

```
FindEq(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  name of an attribute enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string)

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was equal to the specified value

Exceptions

If the attribute was never equal to the specified value, an error value is returned

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was equal to the specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindEq('att1', '-1h', '*', 40)`

  [Return the timestamp of the first time since an hour ago when ' att1' was equal to 40]

- `FindEq('enum_att1', '30-March-17', '*', "On")`

  [Return the timestamp of the first time since the start of day (12am) on March 30th, 2017 when enumeration value 'enum_att1' was equal to "On"]

FindGE

Find the first time within a time interval when an attribute is greater than or equal to a specified value.

Syntax

```
FindGE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was greater than or equal to the specified value.

Exceptions

If the attribute was always less than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was greater than or equal to the specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindGE('att1', 'y', '+2h', 40)`

  [Return the timestamp between midnight and 2:00 am yesterday when 'att1' was first greater than or equal to 40.]

FindGT

Find the first time within a time interval when an attribute is greater than a specified value.

Syntax

```
FindGT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **starttime** within the specified interval when the attribute was greater than the specified value.

Exceptions

If the attribute was never greater than the specified value, an error is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was greater than he specified value.
- If **endtime** is earlier than **starttime**, the time interval is searched backwards.

Example

- `FindGT('att1', 't', '*', 40)`

  [Return the timestamp of the first time after midnight today when 'att1' was greater than 40]

- `FindGT('att1', '-1h', '*', TagVal('att1', 'y+18h'))`

  [Return the timestamp of the first time since an hour ago when the value of 'att1' was greater than its value at 6pm yesterday]

FindLE

Find the first time within a time interval when an attribute is less than or equal to a specified value.

Syntax

```
FindLE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **starttime**

- **x**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to *starttime* within the specified interval when the attribute was less than or equal to the specified value.

Exceptions

If the attribute was always greater than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was equal to the specified value.
- If *endtime* is earlier than *starttime*, the time interval is searched backwards.

Example

- `FindLE('att1', 't', '*', 40)`

  [Return the timestamp of the first time after midnight today when *att1* was less than or equal to 40.]

FindLT

Find the first time within a time interval when an attribute is less than a specified value.

Syntax

```
FindLT(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **`starttime`**

- **`x`**

  Must be an integer or real number or digital state (character string), the value to search for

Returns

The timestamp closest to **`starttime`** within the specified interval when the attribute was less than the specified value.

Exceptions

If the attribute was less than the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was lower than the specified value.
- If **`endtime`** is earlier than **`starttime`**, the time interval is searched backwards.

Example

- `FindLT('att1', 'y', 't', 40)`

  [Return the timestamp of the first time between midnight yesterday and midnight today that 'att1' was less than 40.]

- `FindLT('att1', '-1h', '*', TagVal('att1', 'y+18h'))`

  [Return the timestamp of the first time since an hour ago when the value of 'att1' was less than its value at 6pm yesterday]

FindNE

Find the first time within a time interval when an attribute is not equal to a specified value. "No Data" events will be skipped.

Syntax

```
FindNE(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute enclosed in single quotation marks.

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `"-3h"`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `"+1h"`) in reference to an absolute **`starttime`**

- **`x`**

  The value to search for; must be an integer or real number or digital state (character string).

Returns

The timestamp closest to **`starttime`**, within the specified interval, for which the attribute was not equal to the specified value. "No Data" events will be skipped.

Exceptions

If the attribute was always equal to the specified value, an error value is returned.

Notes

- If needed, interpolation is done between attribute values to determine when the attribute was not equal to the specified value.
- If **`endtime`** is earlier than **`starttime`**, the time interval is searched backwards.

Example

- `FindNE('att1', '-1h', '*', 40)`

[Return the timestamp of the first time since an hour ago when ' att1' was not equal to 40]

- `FindNE('enum_att1', '30-March-17', '*', "On")`

  [Return the timestamp of the first time since the start of day (12am) on March 30th, 2017 when enumeration value 'enum_att1' was not equal to "On"]

FirstValue

Get the first value in an array that satisfies a given condition.

Syntax

```
FirstValue(array [, condition($val)])
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

The first value that satisfies the given condition

Exceptions

None

Notes

None

Example

- `FirstValue(Data)`

  [Return the first value from an array named **`Data`**]

- `FirstVal(Data, $val < 90)`

  [From the array named **`Data`**, return the first value in the array that is less than 90]

Float

Convert a string to a number.

Syntax

```
Float(x)
```

Arguments

- **x**

  A string or a number, or an attribute whose value evaluates to a number at time of evaluation

Returns

A number for a numeric string. If *x* is already a number, *x* is returned

Exceptions

If *x* is not a number or a numeric string, returns `Calc Failed`

Notes

Unit of measure of the argument, if it exists, is carried over to the result. **`Float`** also takes *timespan* and *boolean* as argument. Note, however, that **`Float`** only converts *timespan* format to the number of seconds from 12:00am Jan 1, 1970.

Example

- `Float("12.3")`

  [Converts string to a number and returns 12.3]

- `Float(12.3)`

  [Returns 12.3]

- `Float('*')`

  [Return the total number of seconds passed since Jan 1, 1970]

Floor

Return the nearest integer less than or equal to *x*.

Syntax

```
Floor(x)
```

Arguments

- **x**

  Integer or a real number, or an attribute whose value evaluates to a number at time of evaluation

Returns

The nearest integer less than or equal to *x*

Exceptions

Returns an error if *x* is a digital state, time expression, time interval or a string

Notes

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Floor(3.14)`

  [Returns 3]

- `Floor(-3.14)`

  [Returns -4]

- `Floor(TagVal('att1', '*'))`

  [Return the nearest integer less than or equal to the current value of 'att1']

Format

Convert a number to string according to a format expression.

Syntax

```
Format(x, format [,culture])
```

Arguments

- **x**

  An integer or a real number

- **format**

  Format-control string, similar to that used by the C language function **Sprintf**

- **culture**

  Optional. Culture name string, enclosed in double quotation marks. Example values include "de-DE", "ja-jp", "pt-br", "zh-cn". Use "" to specify the culture-invariant setting (the default setting).

Returns

A formatted string

Example

- `Format(1234.567, "%10.2f", "de-DE")`

  [Returns "1234,57", a formatted string]

- `Format(45, "%3.3d")`

  [Returns "045"]

Frac

Return the fractional part of a real number. Returns 0 for integers.

Syntax

```
Frac(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The fractional part of *x*

Exceptions

If *x* is not an integer or real number, returns an error value

Notes

By definition: `Int(x) + Frac(x) = x`

Unit of measure of the argument, if it exists, is carried over to the result

Example

- `Frac(1)`

  [Returns 0]

- `Frac(1.3)`

  [Returns 0.3]

- `Frac(TagVal('att1', '*'))`

  [Return the fractional part of the value of 'att1' at current time.]

HasChanged

Return `True` if an attribute has had any event in the specified time period; otherwise return `False`.

Syntax

```
HasChanged(attname, t1)
```

Arguments

- **attname**

  The name of an attribute enclosed in single quotation marks

- **t1**

  The start of a time period ending at execution time; can be any relative time expression in single quotation marks (such as 't - 4h')

Returns

`True` if *attname* has any event in the specified time period; otherwise returns `False`

Example

- `HasChanged('att1', 't-4h')`

[Returns `True` if 'att1' received any updates since 8 p.m. last night, regardless of whether the update changed the actual attribute value]

HasValueChanged

Determine if the value of an attribute or expression has changed since last evaluated during an analysis.

Syntax

```
HasValueChanged(x)
```

Arguments

- **x**

  attribute or expression

Returns

`True` if the value of *x* has changed since last evaluated during the analysis; otherwise returns `False`.

If *x* is an expression that this function has not yet evaluated during the analysis, then the function returns `False`.

If *x* is an attribute that this function has not yet evaluated during the analysis, then the function finds the previous recorded value and compares the current value to that previous value. If there is no previous value, then the function returns `False`.

Notes

If PI Analysis Service restarts during an analysis, any previous evaluations are lost.

Example

- `HasValueChanged('att1')`

  [Returns `True` if the value of *att1* has changed since last evaluated during the analysis]

- `HasValueChanged('att1'+'att2')`

  [Returns `True` if the value of sum of *att1* and *att2* has changed since last evaluated during the analysis]

Hour

Extract the hour from a time expression.

Syntax

```
Hour(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes.

Returns

The hour of time, in the range 0-23.

Exceptions

None

Example

- `Hour('*')`

  [Return the hour portion of current time.]

- `Hour('Saturday')`

  [Returns 0.]

- `Hour(FindEq('att1', '-1d', '*', 50))`

  [Return the hour of the time when the value of 'att1' was first equal to 50 in the past 24 hours.]

IF

Operator that introduces the condition in **IF-THEN-ELSE** statement. Return the first of two specified values if the conditional expression is true. Otherwise, return the second value.

Syntax

```
IF (expression) THEN (x) ELSE (y)
```

Arguments

- **expression**

  Any expression that evaluates to true or false

- **x, y**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return 'the value of 'att2'; otherwise return the value of 'att3']

IN

Return **True** if a value is included in a set or a range of inclusive values.

Syntax

```
x In (y1 [, ... yn])
```

```
x In (y1...yn)
```

Arguments

- **x**

  numeric value or string

- **y_1 [, ... y_n]**

  a set of numeric or string values

- **y_1 ... y_n**

  a range of numeric values

Returns

True when *x* is included in a set or a range of values

Exceptions

None

Example

- `1 In (9, 10)`

  [Returns `False` since 1 is not included in a set of 9 and 10]

- `2 In (1...10)`

  [Returns `True` since 2 is included in a range of 1 to 10]

InStr

Return the location within a string where a sub-string match is first found.

Syntax

```
InStr([start,] s1, s2 [,caseSensitive])
```

Arguments

- **start**

  Optional: An integer specifying which character in *s1* to start the comparison. Must be larger than or equal to 1.

- **s1, s2**

  Two strings to be compared.

- **`caseSensitive`**

  Optional: A flag indicating if the comparison is case-sensitive. If 0 (the default) the comparison is case-insensitive, if 1, the comparison is case-sensitive.

Returns

0 if *s2* is not a sub-string of *s1* starting from the start position; otherwise, the location of character where *s2* first matches the characters in *s1* from the start position.

Exceptions

Wildcard characters are not treated as wildcards.

Example

- `InStr("What", "At")`

  [Returns 3]

- `InStr("What What What", "What")`

  [Returns 1]

- `InStr("what", "At", 1)`

  [Returns 0]

- `InStr(4, "what", "At")`

  [Returns 0]

- `InStr('att1', "Error")`

  [Returns 1 if the value of *att1* is "Error"]

Int

Return the integer part of an integer or real number.

Syntax

```
Int(x)
```

Arguments

- **x**

  Number, Boolean, numeric string, timespan, or an attribute whose value evaluates to a number at time of evaluation

Returns

If *x* is a number, the integer part of *x* is returned. If *x* is a string, it is first converted into a number. If *x* is a *timespan*, the number of seconds from 12:00am January 1, 1970 is returned.

Exceptions

Returns error if the variable is not assigned

Notes

`Int('*')` returns UTC time in seconds from 12:00am January 1, 1970.

Example

- `Int('att1')`

  [Return the integer part of the value of 'att1' at current time.]

- `Int('*'-'t')`

  [Return how many seconds have passed since the start of day today.]

- `Int(2.1)`

  [Returns 2.]

- `Int("2.1")`

  [First converts the string to a number and returns 2.]

- `Int(true)`

  [Returns 1.]

InterpolatedValues

Obtain interpolated values over a specified time range using the specified time step.

Syntax

```
InterpolatedValues(attname, starttime, endtime, timestep)
```

Arguments

- **`attname`**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as '-3h') in reference to an absolute endtime

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as '+1h') in reference to an absolute starttime

- **`timestep`**

  a time interval enclosed in single quotes (such as '+1m' or '-10s') representing the incremental change in time between the specified range to obtain the interpolated values

Returns

An array of the values obtained within the specified range in intervals of the time step

Exceptions

If the attribute does not support range calls or interpolated values of range calls, the function returns an error indicating as such

Notes

- If the **`starttime`** is earlier than the **`endtime`**, the resulting values will be in time-ascending order, otherwise they will be in time-descending order.
- When a positive timestep is specified, the interval calculation begins at the earliest bounding time in the time range and applies the interval repeatedly in the time-ascending direction to generate the calculation intervals.
- If a negative timestep is specified, the interval calculation begins at the latest bounding time in the time range and applies the interval repeatedly in the time-descending direction to generate the calculation intervals.
- Note that the order of values returned will still be reflected by the time range, regardless of the timestep sign.

Example

- `InterpolatedValues('att1', 't', '+1h', '+10m')`

  [Return the values of 'att1' between 12:00 and 1:00am today in 10-minute intervals]

- `InterpolatedValues('att1', 't', '+1h', '-14m')`

  [Return the values of 'att1' between 12:00 and 1:00am today in 14-minute intervals, anchoring from 1:00am (that is, 12:04,12:18,12:32,12:46,1:00)]

IsSet

For an attribute associated with a PI point, determine if the point is annotated, substituted, or questionable.

Syntax

```
IsSet(attname, select)
```

Arguments

- **`attname`**

  An attribute associated with a PI point of value type integer, real number, enumeration value, or string

- **`select`**

  A string but only the first character is considered: "a" for annotate, "s" for substituted and "q" for questionable. Not case-sensitive.

Returns

`True` or `False`

Exceptions

None

Example

- `IsSet('att1', "a")`

  [Returns `True` if the value of 'att1' is annotated]

- `IsSet('att1', "s")`

  [Returns `True` if the value of 'att1' is substituted]]

- `IsSet('att1', "q")`

  [Returns `True` if the value of 'att1' is questionable]

LastValue

Get the last value in an array that satisfies a given condition.

Syntax

```
LastValue(array [, condition($val)])
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`condition($val)`**

  any supported PE statement, as a function of **`$val`**, that returns a true or false. **`$val`** is a placeholder for each value in the array

Returns

The last value that satisfies the given condition

Exceptions

None

Notes

None

Example

- `LastValue(Data)`

  [Return the last value from an array named **`Data`**]

- `LastValue(Data, $val > 90 and $val < 105)`

  [From the array named **`Data`**, return the last value in the array that is greater than 90 and less than 105]

LCase

Convert a string to a lowercase string.

Syntax

```
LCase(s1)
```

Arguments

- **s1**

  string

Returns

A string that has been converted to lowercase

Exceptions

If the argument is not a string, returns an error value.

Example

- `LCase("String")`

  [Returns "string"]

Left

Return a specified number of characters of a string from the left.

Syntax

```
Left(s1, len)
```

Arguments

- **s1**

  String

- **len**

  Integer.

Returns

The first **len** characters of the string, starting from the left.

Exceptions

If the arguments are not of the required types, returns an error.

Example

- `Left("String_att", 3)`

  [Returns "Str"]

Len

Return the length of a string.

Syntax

```
Len(s1)
```

Arguments

- **s1**

  string

Returns

The length of a string

Exceptions

If the argument is not a string, returns an error value.

Example

- `Len("String")`

  [Returns 6]

- `Len('sinusoid')`

  [Returns `Calc Failed`]

LinRegr

Apply linear least squares fitting to two sets of values given by attributes over a specified time range. The output is a one-based array with the values of the slope, intercept and R^2.

Syntax

```
LinRegr(x, y, mode, starttime, endtime [, pctgood])
```

```
LinRegr(y, starttime, endtime [, pctgood])
```

Arguments

- **x**

  an attribute with the first set of time series data (such as PI point data reference) enclosed in single quotes

- **y**

  an attribute with the second set of time series data (such as PI point data reference) enclosed in single quotes

- **mode**

  a number that specifies how to align time-stamped values

  choose from 0, 1 or 2 where:

0 represents the combination of time-stamped values from *x* and *y*

1 represents values from both attributes according to *x*'s time stamps

2 represents values from both attributes according to *y*'s time stamps

- **`starttime`**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute *endtime*

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute *starttime*

- **`pctgood`**

  Optional. Minimum percentage of time during the time range that attribute values must be good.

  You can set *pctgood* as a threshold to ensure that there are sufficient good values to calculate **`LinRegr`**.

Returns

One-based array with the values of slope, intercept and R^2.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value

Notes

Bad values are excluded from **`LinRegr`** calculation

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **`PctGood`** function to find out what percentage of the values is good

Index the slope, intercept and R^2 outputs from [1].

Name	Expression	Value at Evaluation
RegressionData1	`LinRegr('XValues','YValues',2,'t','*')`	[0.27589, 219.05, 0.50575]
RegressionData2	`LinRegr('XValues','YValues',2,'t','*',80)`	[0.27589, 219.05, 0.50575]
Slope	`RegressionData1[1]`	0.27589
Intercept	`RegressionData1[2]`	219.05
RSquared	`RegressionData1[3]`	0.50575

Example

- `LinRegr('att1', 'att2', 0, 't', '+1h', 80)`

  [Return a one-based array of slope, intercept and R^2 based on time-stamped values of 'att1' and 'att2' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `LinRegr('att1', 'att2', 1, 't', '+1h')`

  [Return a one-based array of slope, intercept and R^2 based on time-stamped values of 'att1' between 12:00 and 1:00am today]

- `LinRegr('att1', 't', '+1h', 80)`

  [Return a one-based array of slope, intercept and R^2 of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return an error when minimum *pctgood* is not reached]

- `LinRegr('att1', 't', '+1h')`

  [Return a one-based array of slope, intercept and R^2 of 'att1' between 12:00 and 1:00am today]

Log

Return the natural logarithm (base e = 2.7182818...) of an integer or real number.

Syntax

```
Log(x)
```

Arguments

- **x**

  Must be an integer or real number greater than zero

Returns

The natural logarithm of *x*

Exceptions

If *x* is zero or negative, or not a number, returns an error value

Example

- `Log(14)`

  [Returns 2.6391]

- `Log(TagVal('att1', '14-Dec-16'))`

  [Return the natural log of the value of 'att1' at 12:00am on Dec 14, 2016]

Log10

Return the base 10 logarithm of an integer or real number.

Syntax

```
Log10(x)
```

Arguments

- **x**

  Must be an integer or real number greater than zero

Returns

The base 10 logarithm of *x*

Exceptions/Errors

If *x* is zero or negative, or not a number, returns an error value

Example

- `Log10(100)`

  [Returns 2]

- `Log10(TagVal('att1', '14-Dec-16'))`

  [Return the base 10 logarithm of the value of 'att1' at 12:00am on Dec 14, 2016]

Logbase

Return the logarithm of positive numeric value *x* to a specified base *y*.

Syntax

```
Logbase(x,y)
```

Arguments

- **x**

  An integer or real number greater than zero

- **y**

  A positive integer indicating the base of the logarithm

Returns

The logarithm of *x* to base *y*

Exceptions

If *x* is zero or negative, or not a number, returns an error value

Example

- `Logbase(256, 2)`

  [Returns 8]

- `Logbase(1000, 10)`

[Returns 3]

- `Logbase(TagVal('att1', '14-Dec-16'), 16)`

  [Return the logarithm of the value of 'att1' at 12:00am on Dec 14, 2016 to base 16]

LTrim

Remove the leading blanks from a string.

Syntax

```
LTrim(s1)
```

Arguments

- **`s1`**

  string

Returns

A string with leading blanks removed

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `LTrim(" String")`

  [Returns "String"]

- `LTrim("String ")`

  [Returns "String "]

MapData

Apply a function to each value in an array.

Syntax

```
MapData(array, function($val))
```

Arguments

- **`array`**

  a variable representing the array to operate on

- **`function($val)`**

  any supported PE statement, as a function of **`$val`**, that performs an operation. **`$val`** is a placeholder for each value in the array

Returns

The transformed array of all the values

Exceptions

None

Notes

None

Example

- `MapData(Data, (IF BadVal($val) THEN 0 ELSE $val))`

  [From an array named **Data**, return a new array with bad values set to 0]

Max

Return the maximum from a set of values.

Syntax

```
Max(x1, ... xn)
```

Arguments

- **x1, ... xn**

  Arguments or a single array of same value type (integers and real numbers, enumeration values, time expressions, or time intervals)

Returns

The maximum from a set of arguments.. The result is of the same data type as the arguments

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, **Max** returns an error. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs

Example

- `Max(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Find the maximum from these values: 14, value of 'att1' at trigger time, 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `Max('enum_att1', 'enum_att2', 'enum_att3')`

  [At trigger time, find the highest value from an enumeration set and return its name]

- `Max('*', 'y', 'Saturday')`

  [Find the most recent timestamp. Returns current time from the example above]

- `Max('*'-'*-1h', 't+8h'-'y+4h', '*'-'t')`

[Find the longest interval from these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `Max(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Max(Variable1)	9

[Find the maximum of values in an array `Variable1`]

Median

Return the median (middle value) of one or more values.

Syntax

```
Median(x1 [, ... xn])
```

Arguments

- **`x1 [, ... xn]`**

  Arguments and single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The median value of the input argument(s). If there are even number of arguments, the average of the two middle values is returned. Returns an error if the array does not consist of same value type.

Exceptions

Arguments whose run-time values are digital states are ignored. The function must have one or more arguments that evaluate to non-digital states; otherwise, `Median` returns an error value. Also returns an error if the array does not consist of same value type.

Notes

Arguments must be of same value type. The data type of the first argument sets the tone for the rest.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Examples

- `Median(14, 'att1', 14.5, TagVal('att2', '14-Dec-16'))`

[Find the median of these values: 14, the current value of *att1*, 14.5, and the value for *att2* at midnight on Dec 14, 2016]

- `Median('*', 'y', 'Saturday')`

  [Find the median of these timestamps: now, 12:00am yesterday, and 12:00am last Saturday]

- `Median('*'-'*-1h', 't+4h'-'y+8h', 'y+2h'-'20')`

  [Find the median of these time periods: from 1 hour ago to now, from 8:00am yesterday to 4:00am today, and 2:00am yesterday to the start of day (12am) on the 20th of this month]

- `Median(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Median(Variable1)	5

  [Find the median of values in an array `Variable1`]

Mid

Return a sub-string within a string.

Syntax

```
Mid(s1, start [,len])
```

Arguments

- **`s1`**

  string

- **`start`**

  An integer specifying the position of the first character within the string. The first character in the string is number 1.

- **`len`**

  Optional: The maximum length of the returned string. The default is the length of the string.

Returns

len characters of the string to the right of (and including) the first character whose position is specified by start.

Exceptions

If the arguments are not of the required types, an error value is returned. The maximum number of characters that can be returned is 999.

Example

- `Mid("String", 3)`

[Returns "ring"]

- `Mid("String", 3, 2)`

  [Returns "ri"]

Min

Return the minimum from a set of values.

Syntax

```
Min(x1, ... xn)
```

Arguments

- **`x1, ... xn`**

  Arguments or a single array of same value type (integers and real numbers, enumeration values, time expressions, or time intervals)

Returns

The minimum from a set of arguments. The result is of the same data type as arguments

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, **`Min`** returns an error. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs

Example

- `Min(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Find the minimum from these values: 14, value of 'att1' at trigger time, 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `Min('enum_att1', 'enum_att2', 'enum_att3')`

  [At trigger time, find the lowest value from an enumeration set and return its name]

- `Min('*', 'y', 'Saturday')`

  [Find the oldest timestamp from a set]

- `Min('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Find the shortest interval from these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `Min(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Min(Variable1)	1

[Find the minimum of values in an array `Variable1`]

Minute

Extract the minute from a time expression.

Syntax

```
Minute(t1)
```

Arguments

- **`t1`**

  A time expression, enclosed in single quotes.

Returns

The minute of time, in the range 0-59.

Exceptions

None.

Example

- `Minute('*')`

  [Extract the minute from current time.]

- `Minute(FindGT('att1', '-1h', '*', 5)`

  [Extract the minute from a timestamp when the value of 'att1' was first greater than 5 in the past hour. Return error if it was never over 5.]

Mod

The modulus operator (mod) returns the remainder from the quotient of two numeric values. For *x* **`Mod`** *y*, this is the remainder from *x*/*y*.

Syntax

```
x Mod y
```

Arguments

- **`x`**

  Any expression that evaluates to a numeric value

- **y**

  Any non-zero numeric value

Returns

Remainder from *x*/*y*

Exceptions

None

Notes

UOM of *x*, if it exists, is carried over to the result

Example

- `11 Mod 3`

  [Returns 2]

Month

Extract the month from a given time expression.

Syntax

```
Month(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The month of time, in the range 1-12.

Exceptions

None

Example

- `Month('*')`

  [Return the current month.]

- `Month(FindEq('att1', '-10d', '*', 5')`

  [Return the month from a timestamp when the value of 'att1' was first equal to 5 in the past 10 days.]

NextEvent

Find the timestamp of the next recorded value after a specified time.

Syntax

```
NextEvent(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The timestamp of the next recorded value after the specified time for the specified attribute

Exceptions

If there is no recorded value after the specified time, then the function returns `No Data`.

Example

- `NextEvent('att1', 't')`

  [Find and return the timestamp of the next recorded value of 'att1' since the start of day (12am) today]

NextVal

Find the next recorded value after a specified time.

Syntax

```
NextVal(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The next recorded value after the specified time for the specified attribute

Exceptions

If there is no recorded value after the specified time, the function returns an error.

Example

- `NextVal('att1', 't')`

  [Find and return the next recorded value of 'att1' since the start of day (12am) today]

Noon

Return a timestamp for noon on the day of a given time expression.

Syntax

```
Noon(t1)
```

Arguments

- **t1**

  A time expression enclosed in single quotes

Returns

A timestamp corresponding to noon of the day of the input time

Exceptions

None

Notes

This function is useful for establishing a unique clock time independent of the length of particular days.

Example

- `Noon('*')`

  [Return the timestamp for noon of the day when 'att1' was first equal to 50 in the past 3 days.]

- `Noon(FindEq('att1', '-3d', '*', 50))`

  [Return the timestamp for noon of the day when 'att1' was first equal to 50 in the past 3 days.]

NoOutput

Do not write current calculation result.

Syntax

```
NoOutput()
```

Arguments

None

Notes

It is important to include the parentheses after this function (use `NoOutput()` instead of `NoOutput` as `NoOutput` is an invalid syntax). This function applies only to the current calculation.

Example

- `If 'att1' < 100 or 'att1' > 200 then 'att1' else NoOutput()`

Normalrnd

Return a random number that maps the normal distribution curve using a specified mean and standard deviation.

Syntax

```
Normalrnd(x, y)
```

Arguments

- **x**

  A real number specifying the mean of the normal distribution curve

- **y**

  A real number specifying the standard deviation of the normal distribution curve

Returns

A random number that maps the normal distribution curve with a specified mean and standard deviation

Exceptions

None

Example

- `Normalrnd(300, 2.5)`

NOT

Return the negation of an expression. Return **True** if the truth value of an expression is false and **False** if the truth value of an expression is true.

Syntax

```
NOT expression
```

Arguments

- **expression**

  Any expression that evaluates to true or false

Returns

True when the expression is false and **False** when the expression is true

Exceptions

None

Example

- `NOT (1 In (1...10))`

  [Returns False since (1 In (1…10)) is true.]

NumOfChanges

Return the number of changes in value for an attribute within a specified time range.

Syntax

```
NumOfChanges(attname, starttime, endtime)
```

Arguments

- **attname**

  The name of an attribute.

- **starttime**

  A time expression that specifies the start of a time range, or a time span (such as '-3h') that specifies the start time relative to *endtime*; entries must be enclosed in single quotation marks

- **endtime**

  A time expression that specifies the end of a time range, or a time span (such as '+3h') that specifies the end time relative to *startime*; entries must be enclosed in single quotation marks

Returns

The count of changes in value for *attname* in the specified time range

Example

- `NumOfChanges('att1', 't', '*')`

  [Return the number of times the value of 'att1' changed since midnight until now]

OR

The logical disjunction of two expressions that returns **True** if either expression is true and **False** if both are false.

Syntax

```
expression1 OR expression2
```

Arguments

- **expression1**

  Any expression that evaluates to true or false

- **expression2**

  Any expression that evaluates to true or false

Returns

True if either expression is true and **False** if both are false

Exceptions

None

Example

- `('att1' > 50) OR ('att2' = "good")`

ParseTime

Translate a PI time expression to a timestamp. Use regular time expression inside single quotes for better performance.

Syntax

```
ParseTime(s1)
```

Arguments

- **s1**

  A character string or an array of strings in PI time format, enclosed in double quotes

Returns

The timestamp corresponding to **s1**

Exceptions

If **s1** is not a character string, or if there is a syntax error, returns an error value .

If the array does not consist of strings in a time format, returns an error.

Notes

Use of the time expression '01-Jan-2018' over the string "01-Jan-2018" is strongly recommended.

Example

- `ParseTime(Concat("12", "-31", "-16"))`

  [Returns 12/31/2016 12:00:00 AM, which is the same as '12/31/16']

- `ParseTime("14-Dec-16")`

  [Renders the same result as '14-Dec-16'. Use only when string operations are necessary]

- `ParseTime("*")`

  [Renders the same result as '*'. Use only when string operations are necessary]

- `ParseTime(Variable1)`

Note:

Name	Expression	Value at Evaluation
Variable1	'TimeStringArray'	[t, t+3h, t+14h]
Variable 2	ParseTime(Variable1)	[3/14/2018 12:00:00 AM, 3/14/2018 3:00:00 AM, 3/14/2018 2:00:00 PM]

[Renders the array results in a proper PI Time format, assuming today is 3/14/18]

PctGood

Find the time percentage, over a specified time range, for which the attribute had good values.

Syntax

```
PctGood(attname, starttime, endtime)
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

Returns

An integer or real number from 0 to 100: the percentage of the specified time for which the attribute had good values

Example

- `PctGood('att1', 't', '+1h')`

  [Return the time percentage between 12:00 and 1:00am today for which 'att1' had good values]

- `PctGood('att1', '-1h', '*')`

  [Return the time percentage between now and hour ago today for which 'att1' had good values]

Pi

Returns the value for pi.

Syntax

```
Pi()
```

Arguments

-

Returns

The value for pi

Exceptions

None.

Example

- `Pi()`

Poisson

Return a random number that maps a Poisson distribution with a specific mean.

Syntax

```
Poisson(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

A random number that maps a Poisson distribution with a specified mean

Exceptions

None

Example

- `Poisson(15)`

Poly

Return the polynomial $c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$.

Syntax

```
Poly(x, c0 [, ... cn])
```

Arguments

- **x**

  Variable. An integer or real number

- **c0 [, ... cn]**

  Coefficients. There must be at least one coefficient. All must be numbers.

Returns

The value of the polynomial

Exceptions

If *x* or any coefficient is not an integer or real number, **Poly** returns an error value

Example

- `Poly(3, 4, 5)`

  [Returns 19]

- `Poly('att1', 2, 3)`

Pow

Return **x** raised to the power **y**.

Syntax

```
Pow(x, y)
```

Arguments

- **x**

  numeric value

- **y**

  numeric value

Returns

The value of *x* raised to the power *y*

Exceptions

None

Notes

- The **Pow** function only works in PI Asset Framework (PI AF) analytics expression functions. Use caret (^) operator for same effect in PI Data Archive performance equations
- Both `Pow(2, 3)` and 2^3 are acceptable in PI AF analytics expression functions

Example

- `Pow(2, 3)`

  [Returns 8]

- `Pow(2.5, 3)`

  [Returns 15.625]

- `Pow(TagVal('att1', '*'), 8)`

  [Return the current value of 'att1' raised to the power 8]

PrevEvent

Find the timestamp of the last recorded value before a specified time.

Syntax

```
PrevEvent(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The timestamp of the last recorded value before the specified time for the specified attribute

Exceptions

If there is no recorded value before the specified time, the function returns an error.

Example

- `PrevEvent('att1', '*')`

  [Find the timestamp of the last recorded value of 'att1' before current time]

PrevVal

Find the last recorded value before a specified time.

Syntax

```
PrevVal(attname, t1)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks

- **t1**

  A time expression

Returns

The last recorded value before the specified time for the specified attribute

Exceptions

If there is no recorded value before the specified time, the function returns an error.

Example

- `PrevVal('att1', '*')`

  [Find and return the last recorded value of 'att1' before current time]

- `PrevVal('att1', '15-Mar-17')`

  [Find and return the last recorded value of 'att1' before start of day (12am) March 15th, 2017]

PStDev

Return the *population standard deviation* for a population of one or more values.

Syntax

```
PStDev(x1 [, ... xn])
```

Arguments

- **`x1, ... xn`**

  Arguments and a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The *population standard deviation* for the arguments. Returns a numeric value if the arguments are numbers. For arguments that are time expressions (time or time period), a number indicating a time period expressed in seconds is returned.

The population standard deviation of a population x1, ..., xn is

$$\sqrt{\frac{\sum (x_i - \mu)^2}{n}}$$

where μ is the mean of the arguments; that is,

$$\frac{\sum x_i}{n}$$

Exceptions

Arguments whose run-time values are digital states are ignored. If all values are digital states, an error value is returned. Returns an error if the array does not consist of same value type.

Notes

`PStDev` uses every value in a population to calculate the population standard deviation. However, it is common, especially for a large population, to estimate standard deviation from a sample of the population. **`SStDev`** uses a set of sample values to calculate sample standard deviation, which approximates the population standard deviation.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `PStDev(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

  [Return the "population standard deviation" of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `PStDev('*', 'y', 'Saturday')`

  [Return the "population standard deviation" of these timestamps.]

- `PStDev('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Return the "population standard deviation" of these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `PStDev(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	PStDev(Variable1)	2.8284

  [Find the "population standard deviation" of values in an array `Variable1`]

Rand

Return a random number between 0 and 1. For specified *x* and *y* values, return a random number between x - y/2 and x + y/2.

Syntax

```
Rand(x, y)
```

Arguments

- **x**

  A real number specifying the center point of the range

- **y**

  A real number specifying the size of the range.

If no arguments are specified, the default range is from 0 to 1.

Returns

A random number between 0 and 1. For specified *x* and *y* values, returns a random number between x- y/2 and x + y/2.

Exceptions

None

Example

- `Rand()`
- `Rand(500, 250)`

Range

Find the difference between the maximum and minimum values for an attribute during a specified time range.

Syntax

```
Range(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`pctgood`**

  Optional: Minimum time percentage, over the given time range, that values for the attribute must be good.

Returns

The difference between the attribute's maximum and minimum values during the specified time.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached in the given time range, an error is returned.

Notes

Bad values are excluded from **Range** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `Range('att1', 't', '+1h')`

  [Return the difference between the maximum and minimum values for 'att1' between 12:00 and 1:00am today.]

- `Range('att1', 't', '+1h', 80)`

  [Return the difference between the maximum and minimum values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

Rate

Return the difference over time between the current value and a previously evaluated value of an attribute or (time) expression.

Syntax

```
Rate(x)
```

Arguments

- **x**

  An attribute or (time) expression that resolves to a numeric value

Returns

- The calculated difference, using the value of the attribute, numeric, or timestamp value of the expression parameter, according to this formula: (current value – previously evaluated value) / (current timestamp – timestamp from previous evaluation).

Example

- `Rate(‘att1’)`
- `Rate('sinusoid')`
- `Rate(DaySec(‘*’) + 1)`

RecordedValues

Obtain values within a particular time range based on user-specified boundary type.

Syntax

```
RecordedValues(attname, starttime, endtime [, boundarytype])
```

Arguments

- **attname**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  a time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as '-3h') in reference to an absolute endtime

- **`endtime`**

  a time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as '+1h') in reference to an absolute starttime

- **`boundarytype`**

  a string indicating the data retrieval behavior at the end points of a specified time in double quotes:

  "Inside" - return the nearest recorded values inside the requested time range as the first and last values

  "Outside" - return the nearest recorded values on the outside of the requested time range as the first and last values

  "Interpolated" - create an interpolated value at the end points of the requested time range if a recorded value does not exist at that time

  > **Note:**
  > If unspecified, the default is the "Inside" mode.

Returns

An array of the values obtained within the specified range

Exceptions

If the attribute does not support range calls, the function returns an error with indication

Notes

- If the **`starttime`** is earlier than the **`endtime`**, the resulting values will be in time-ascending order, otherwise they will be in time-descending order.
- When no values are available but you specified "Outside" or "Interpolated" boundary types, a special value (**NoData**) is returned.
- This function retrieves up to 500,000 values at a time.

Example

- `RecordedValues('att1', 't+2h', 't+8h')`

  [Return the values of 'att1' between 2:00 and 8:00am today using the default "Inside" retrieval mode]

- `RecordedValues('att1', 't', '+1h', "Outside")`

  [Return the values of 'att1' between 12:00 and 1:00am today using the "Outside" retrieval mode]

RecordedValuesByCount

Obtain a specified number of values beginning at the requested start time in the direction specified.

Syntax

```
RecordedValuesByCount(attname, starttime, count [, timedirection, boundarytype])
```

Arguments

- **`attname`**

  attribute of time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  a time expression representing the beginning of request

- **`count`**

  the number of stored (recorded) values to return. The value must be greater than zero

- **`timedirection`**

  a string indicating the time direction in data retrieval enclosed in double quotes:

  "Backward" - begin at the start time and move backward in time. Values will be returned in time-descending order

  "Forward" - begin at the start time and move forward in time. Values will be returned in time-ascending order

  > **Note:**
  > If unspecified, the default is the "Backward" mode.

- **`boundarytype`**

  a string indicating the data retrieval behavior at the end point of a specified time in double quotes:

  "Inside" - return the nearest recorded values inside the requested time boundary

  "Outside" - return the nearest recorded values on the outside of the requested time boundary

  "Interpolated" - create an interpolated value at the requested time boundary if a recorded value does not exist at that time

  > **Note:**
  > If unspecified, the default is the "Inside" mode.

Returns

An array of the specified number of values (if available)

Exceptions

If the attribute does not support the calls, the function returns an error with indication

Notes

- When no values are available but you specified "Outside" or "Interpolated" boundary types, a special value (**NoData**) is returned.

Example

- `RecordedValuesByCount('att1', 't+2h', 5)`

  [Return 5 values of 'att1' in time-descending order starting from 2:00am today using the default "Inside" retrieval mode moving backwards in time]

- `RecordedValuesByCount('att1', 't+8h', 3, "Forward", "Outside")`

  [Return the first 3 values of 'att1' since 8am today using the "Outside" retrieval mode]

Remainder

Return the remainder resulting from the division of x by y using the IEEERemainder method. This remainder is equal to x - (y Q), where Q is the quotient of x / y rounded to the nearest integer (if x / y falls halfway between two integers, the even integer is returned). If x - (y Q) is zero, the value +0 is returned if x is positive, or -0 if x is negative. If y = 0, NaN (not a number) is returned.

Syntax

```
Remainder(x, y)
```

Arguments

- **x**

  Any numeric value

- **y**

  Any non-zero numeric value

Returns

Return the remainder resulting from the division of x by y using the IEEERemainder method

> **Note:**
> This is not the same as the remainder returned using the Mod (modulus) operator

Exceptions

None

Notes

Unit of measure of *x*, if it exists, is carried over to the result

Example

- `Remainder(11, 3)`

  [Returns -1]

- `Remainder(10, 3)`

[Returns 1]

Right

Return a specified number of characters of a string from the right.

Syntax

```
Right(s1, len)
```

Arguments

- **s1**

  string

- **len**

  integer

Returns

len characters of the string from the right

Exceptions

If the arguments are not of the required types, an error value is returned.

Example

- `Right("String", 3)`

  [Returns "ing"]

- `Right("String", 20)`

  [Returns "String"]

Round

Round a number or time to the nearest unit.

Syntax

```
Round(x [, unit] [, roundingMode])
```

Arguments

- **x**

  Integer, real number, time expression, or time interval

- **unit**

  Optional. The size of the unit to round to. If *x* is a number, unit must be a number. If *x* is a time expression or time period, unit must be a time expression. If unit is omitted, **Round** rounds to the nearest even integer (for numbers) or second (for time expressions).

- **`roundingMode`**

  Optional. The type of rounding to apply. Using the string "AwayFromZero" applies rounding of midpoint values away from zero. Omission of this parameter, or the use of "ToEven", applies the default banker's rounding.

Returns

The nearest value to *x* which is an integer multiple of unit. Returns the same data type as *x*. For more details, see the following examples.

Exceptions

If *x* is a string, or if unit is of the wrong data type, an error value is returned.

Notes

The system uses a banker's rounding which rounds a number to the nearest even number.

If *x* is time and unit is omitted, this routine has no effect: times are accurate only to a second.

Unit of measure of the argument, if it exists, is carried over to the result.

Example

- `Round(12.499)`

  [Returns 12 (rounded to the nearest integer)]

- `Round(12.5)`

  [Returns 12 (rounded to the nearest even integer)]

- `Round(12.5, "AwayFromZero")`

  [Returns 13 (rounded up to the nearest integer)]

- `Round (13.5)`

  [Returns 14 (rounded to the nearest integer)]

- `Round(13.45, 0.1)`

  [Returns 13.4 (rounded to the nearest even number in one decimal place)]

- `Round(13.45, 0.1, "AwayFromZero")`

  [Returns 13.5 (rounded up to the nearest number in one decimal place)]

- `Round(1285, 10)`

  [Returns 1280 (rounded to the nearest even 10)]

- `Round(1285, 10, "AwayFromZero")`

  [Returns 1290]

- `Round(1285, 10, "ToEven")`

  [Returns 1280 (rounded to the nearest even 10)]

- `Round('14-Dec-16 11:47:16')`

  [Returns 14-Dec-16 11:47:16 (rounded to the nearest second)]

- `Round('14-Dec-16 12:30', '+1h')`

  [Returns 14-Dec-97 12:00:00 (rounded to the nearest even hour)]

- `Round('18:47:15'-'15:00:09')`

  [Returns 03:47:06 (rounded to the nearest second)]

- `Round('18:45:40'-'15:15:10','+1m')`

  [Returns 03:30:00 (rounded to the nearest minute)]

> **Note:**
>
> **Round** to the nearest day results in a timestamp of the closest day in UTC time and not local time.

Roundfrac

Round a numeric value x to y decimal places.

Syntax

```
Roundfrac(x, y [,roundingMode] )
```

Arguments

- **x**

  Must be an integer, real number or time expression

- **y**

  Integer that specifies the number of decimal places to round *x* to

- **roundingMode**

  Optional. The type of rounding to apply. Using the string "AwayFromZero" applies rounding of midpoint values away from zero. Omission of this parameter, or the use of "ToEven", applies the default banker's rounding

Returns

The value of x rounded to y decimal places. Returns the same data type as *x*. For more details, see the examples

Exceptions

If *x* is a string, or if unit is of the wrong data type, returns an error value

Notes

The system uses a banker's rounding which rounds a number to the nearest even number.

Unit of measure of the argument, if it exists, is carried over to the result.

If *x* is time and unit is omitted, this routine has no effect: times are accurate only to 1 second.

Example

- `Roundfrac (Pi(), 4)`

[Returns 3.1416]

- `Roundfrac (0.005, 2)`

  [Returns 0]

- `Roundfrac (0.005, 2, "ToEven")`

  [Returns 0.01]

- `Roundfrac (0.005, 2, "AwayFromZero")`

  [Returns 0.01]

- `Roundfrac (TagVal('att1'), 2)`

  [Return the value of 'att1' rounded to 2 decimal places]

RTrim

Trim trailing blanks from a string.

Syntax

```
RTrim(s1)
```

Arguments

- **s1**

  string

Returns

The source string with trailing blanks removed.

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `RTrim("String ")`

  [Returns "String"]

- `RTrim(" String")`

  [Returns " String"]

Sec

Return the trigonometric secant of a number.

Syntax

```
Sec(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The secant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sec(1)`

  [Returns 1.8508]

- `Sec(1.1)`

  [Returns 2.2046]

- `Sec('att1')`

  [Return the trigonometric secant of the value of 'att1' at time of evaluation]

Sech

Return the hyperbolic secant of a number.

Syntax

```
Sech(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic secant of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sech(1)`

  [Returns 0.64805]

- `Sech(1.1)`

  [Returns 0.59933]

- `Sech('att1')`

  [Return the hyperbolic secant of the value of 'att1' at trigger time]

Second

Extract the second from a time expression.

Syntax

```
Second(t1)
```

Arguments

- **t1**

  A time expression enclosed in single quotes.

Returns

The second of time, in the range 0-59.

Exceptions

None.

Example

- `Second('*')`

  [Return the second from current time.]

- `Second(FindEq('att1', '-1m', '*', 2)`

  [Return the second from a timestamp when the value of 'att1' was first equal to 2 in the past minute.]

SecSinceChange

Return the time in seconds since an attribute value was updated.

Syntax

```
SecSinceChange(attname)
```

Arguments

- **attname**

  The name of an attribute, enclosed in single quotation marks.

Returns

The number of seconds since the last update of the value for an attribute.

Example

- `SecSinceChange('att1')`

  [Return the number of seconds since the value of 'att1' was updated]

Sgn

Return an indicator of the numerical sign of a number.

Syntax

```
Sgn(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

-1 if x < 0; 0 if x = 0; 1 if x > 0

Exceptions

If *x* is not an integer or real number, returns an error value

Example

- `Sgn(1.1)`

  [Returns 1]

- `Sgn(0)`

  [Returns 0]

- `Sgn(-0.1)`

  [Returns -1]

- `Sgn('att1')`

  [Returns 1 if the value of 'att1' is positive, 0 if it equals 0, and -1 if it is negative]

Sin

Return the trigonometric sine of a number.

Syntax

```
Sin(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The sine of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sin(1)`

  [Returns 0.84147]

- `Sin(1.1)`

  [Returns 0.89121]

- `Sin('att1')`

  [Return the trigonometric sine of the value of 'att1' at trigger time]

Sinh

Return the hyperbolic sine of a number.

Syntax

```
Sinh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic sine of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Sinh(1)`

  [Returns 1.1752]

- `Sinh(1.1)`

  [Returns 1.3356]

- `Sinh('att1')`

  [Return the hyperbolic sine of the value of 'att1' at trigger time]

Sqr

Return the square root of a number.

Syntax

```
Sqr(x)
```

Arguments

- **x**

  Must be an integer or real number equal to or greater than zero

Returns

The square root of *x*

Exceptions

If *x* is negative, or is not a number, returns an error value

Example

- `Sqr(2)`

  [Returns 1.4142]

- `Sqr(2.1)`

  [Returns 1.4491]

- `Sqr('att1')`

  [Return square root of the value of 'att1' at trigger time]

StateNo

Translate a digital state into its corresponding enumeration value.

Syntax

```
StateNo(digstate)
```

Arguments

- **digstate**

  An enumeration value

Returns

The offset into the Digital State Set corresponding to *digstate*

Exceptions

If a point is passed as *digstate* that is not a digital point, returns an error value

Notes

A digital state may appear more than once in the digital state table. In this case, the value that **StateNo** returns may vary. If *digstate* is the value of a digital point, **StateNo** returns a code number appropriate for that point

Example

- `StateNo(DigState("No Data"))`

  [Returns 248]

- `StateNo(TagVal('enum_att1', '*-1h'))`

  [Return the digital state number corresponding to the value of 'enum_att1' an hour ago]

SStDev

Return the *sample standard deviation* for two or more arguments that are a sample of a larger population.

Syntax

```
SStDev(x1, x2 [, ... xn])
```

Arguments

- **`x1, ... xn`**

  Arguments and a single array of same value type (integers and real numbers, time expressions, or time intervals)

Returns

The *sample standard deviation* of the arguments. Returns a numeric value if the arguments are numbers. For arguments that are time expressions (time or time period), a number indicating a time period expressed in seconds is returned.

The standard deviation of a sample x1, ... xn is equal to

$$\sqrt{\frac{\sum (x_i - \mu)^2}{n-1}}$$

Where μ is the sample mean; that is,

$$\frac{\sum x_i}{n}$$

Exceptions

Arguments whose run-time values are digital states are ignored. If there are not at least two numeric values, **`SStDev`** returns a zero. Returns an error if the array does not consist of same value type.

Notes

In the rare case where you have the entire population, rather than a sample, you might use the function **`PstDev`**, rather than **`SStDev`**.

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `SStDev(14, 'att1', 14.5, TagVal('att2','14-Dec-16'))`

[Return the "sample standard deviation" of these values: 14, the current value of 'att1', 14.5, and the value for 'att2' at start of day (12:00am) on Dec 14, 2016]

- `SStDev('*', 'y', 'Saturday')`

  [Return the "sample standard deviation" of these timestamps.]

- `SStDev('*'-'*-1h', 't+8h'-'y+4h', 'y'-'20', '*'-'t')`

  [Return the "sample standard deviation" of these time periods: from 1 hour ago to now, from 4:00am yesterday to today at 8am, from 12:00am yesterday to 12:00am on 20th of this month, and from the beginning of day today (12:00am) till now]

- `SStDev(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	SStDev(Variable1)	3.1623

  [Find the "sample standard deviation" of values in an array `Variable1`]

StDev

Find the time-weighted standard deviation of values for an attribute from a specified time range.

Syntax

```
StDev(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`pctgood`**

  Optional: Minimum percentage of time during the specified time range that attribute values must be good.

Returns

The time-weighted standard deviation for attribute values from the specified time range.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, an error is returned.

Notes

Bad values are excluded from **StDev** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `StDev('att1', 't', '+1h')`

  [Return the time-weighted standard deviation of values for 'att1' between 12:00 and 1:00am today.]

- `StDev('att1', 't', '+1h', 80)`

  [Return the time-weighted standard deviation of values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

String

Convert any value to a string.

Syntax

```
String(x)
```

Arguments

- **x**

  Any expression that is of normal PI value type

Returns

The string representing the value argument

Exceptions

None

Example

- `String("Hello, PI user!")`

  [Returns "Hello, PI user!"]

- `String(12.23)`

  [Returns "12.23"]

- `String('sinusoid')`

[Return current value of 'sinusoid' in string]

- `String('*')`

  [Return the current time in string]

TagAvg

Find the time-weighted average of values for an attribute during a specified time range.

Syntax

```
TagAvg(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of time during the time range that attribute values must be good.

Returns

The time-weighted average of attribute values during a specified time range.

Exceptions

If the point has no good values or the *pctgood* minimum is not reached for the given time range, returns an error value.

Notes

Bad values are excluded from **TagAvg** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagAvg('att1', 't', '+1h')`

[Return the time-weighted average of values for 'att1' between 12:00 and 1:00am today.]

- `TagAvg('att1', 't', '+1h', 80)`

  [Return the time-weighted average of values for 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

TagBad

For an attribute associated with a PI point, test the point for an abnormal state at a specified time. If the point is an integer or real number, any digital state is considered abnormal. For digital points, the states that are defined for that point are normal; all others are abnormal.

Syntax

```
TagBad(attname [, t1])
```

Arguments

- **`attname`**

  An attribute associated with a PI point

- **`time`**

  Optional: A time expression. If omitted, current time ('*') is used.

Returns

`False` if the point's state at time is normal, `True` if it is abnormal.

Exceptions

None

Notes

`BadVal` can test any value or expression; **`TagBad`** can only test a PI point.

Example

- `TagBad('att1')`

  [Returns True if PI point for 'att1' does not exist or its value corresponds to system digital state (No Data, for example)]

- `TagBad('enum_att1', '14-Dec-16')`

  [Returns True if PI point for 'enum_att1' does not exist or its value corresponds to system digital state (No Data, for example) at 12:00am on December 14th, 2016]

TagDesc

For an attribute associated with a PI point, retrieve that point's descriptor from the point database.

Syntax

```
TagDesc(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's descriptor

Exceptions

If the PI point does not exist, an error value is returned

Example

- `TagDesc('sinusoid')`

  [Returns **12 Hour Sine Wave**]

- `TagDesc('cdt158')`

  [Returns **Atmospheric Tower OH Vapor**]

TagEU

For an attribute associated with a PI point, retrieve the point's engineering unit string from the point database.

Syntax

```
TagEU(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The PI point's engineering units

Exceptions

If the PI point does not exist, an error values is returned

Notes

Returns blank when PI point lacks an engineering unit

Example

- `TagEU('cdt158')`

  [Returns **DEG.C**]

TagExDesc

For an attribute associated with a PI point, retrieve the point's extended descriptor from the point database.

Syntax

```
TagExDesc(attname)
```

Arguments

- **attname**

  An attribute with a PI point data reference, enclosed in single quotes

Returns

The PI point's extended descriptor

Exceptions

If the PI point does not exist, an error values is returned

Notes

Returns blank when PI point lacks extended descriptor

Example

- `TagExDesc('cdt158')`

  [Returns blank since the point lacks extended descriptor]

TagMax

Find the maximum value of an attribute during a specified time range.

Syntax

```
TagMax(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of the time range that the attribute values must be good.

Returns

The attribute's maximum value during the specified time range.

Exceptions

If the attribute has no good values or the *pctgood* minimum is not reached for the given time range, an error value is returned.

Notes

Bad values are excluded from **`TagMax`** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **`PctGood`** function to find out what percentage of the values is good.

- `TagMax('att1', 't', '+1h')`

  [Return the maximum value of 'att1' between 12:00 and 1:00am today.]

- `TagMax('att1', 't', '+1h', 80)`

  [Return the maximum value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

- `TagMax('att1', '14-Dec-16', '15-Dec-16')`

  [Return the maximum value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagMean

Find the average (mean) of values for an attribute during a specified time range.

Syntax

```
TagMean(attname, starttime, endtime [, pctgood])
```

Arguments

- **`attname`**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **pctgood**

  Optional: Minimum percentage of the specified time range that attribute values must be good.

Returns

The attribute's average value over the given time. Notice that the average is not time-weighted.

Exceptions

If the attribute has no good values or the **pctgood** minimum is not reached for the specified time range, an error value is returned. Unlike some other summary functions, **TagMean** does not interpolate any value on the boundary. Thus, if there is no value in the specified interval, an error value is returned.

Notes

Bad values are excluded from **TagMean** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagMean('att1', 't', '+1h')`

  [Return the average (mean) value of 'att1' between 12:00 and 1:00am today.]

- `TagMean('att1', 't', '+1h', 80)`

  [Return the average (mean) value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good.]

- `TagMean('att1', '14-Dec-16', '15-Dec-16')`

  [Return the average (mean) value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagMin

Find the minimum value of an attribute during a specified time range.

Syntax

```
TagMin(attname, starttime, endtime [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **pctgood**

  Optional: Minimum percentage of the specified time range that the attribute values must be good.

Returns

The minimum value for the attribute during the specified time range.

Exceptions

If no good attribute values exist or if the *pctgood* minimum is not reached for the specified time range, an error value is returned.

Notes

Bad values are excluded from **TagMin** calculation.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the values is good.

Example

- `TagMin('att1', 't', '+1h')`

  [Return the minimum value of 'att1' between 12:00 and 1:00am today.]

- `TagMin('att1', 't', '+1h', 80)`

  [Return the minimum value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

- `TagMin('att1', '14-Dec-16', '15-Dec-16')`

  [Return the minimum value of 'att1' between 12:00am 12/14/2016 and 12:00am 12/15/2016.]

TagName

For an attribute associated with a PI point, get a point's name from the point database.

Syntax

```
TagName(attname)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The name of the PI point associated with *attname*

Exceptions

If the PI point does not exist, an error value is returned

Example

- `TagName('sinusoid')`

  [Returns **SINUSOID**]

- `TagName('cdt158')`

  [Returns **CDT158**]

TagNum

For an attribute associated with a PI point, get a point's number from the point database.

Syntax

```
TagNum(s1)
```

Arguments

- **`string`**

  The name of an attribute with a PI point data reference, enclosed in double quotes

Returns

The point's number

Exceptions

If the point does not exist, returns an error value

Example

- `TagNum("sinusoid")`

  [Returns 1]

- `TagNum("cdt158")`

  [Returns 3]

TagSource

For an attribute associated with a PI point, get a point's **point source** string from the point database.

Syntax

```
TagSource(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's **point source** string

Exceptions

If the point does not exist, an error value is returned

Example

- `TagSource('sinusoid')`

  [Returns R]

- `TagSource('cdt158')`

  [Returns R]

TagSpan

For an attribute associated with a PI point, get a point's span from the point database.

Syntax

```
TagSpan(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's span. If the point's type is **`digital`**, returns an integer whose value is the number of digital state defined for the point.

Example

- `TagSpan('sinusoid')`

  [Returns 100]

- `TagSpan('cdt158')`

  [Returns 200]

TagTot

Find the totalized value (time integral) of an attribute's values over a specified time range, optionally converting the totalized value from its current unit of measure (UOM) to a specified UOM in same class.

Syntax

```
TagTot(attname, starttime, endtime [, toUnit] [, pctgood])
```

Arguments

- **attname**

  attribute with time series data (such as PI point data reference) enclosed in single quotes

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **toUnit**

  Optional: The UOM to which the totalized value is converted. The specified UOM must be compatible with the UOM of the input attribute.

- **pctgood**

  Optional: Minimum percentage of the specified time range that the attribute's values must be good.

Returns

The totalized value (time integral) of an attribute's values over a specified time range.

Exceptions

- If the attribute has no good values or the **pctgood** minimum is not reached for the given time range, an error value is returned.
- If the specified UOM for conversion is incompatible with the UOM of the input attribute, an error value is returned. For example, if the UOM of the input attribute is gallons/minute, and the UOM for conversion is "m" (meters), then an error is returned.

Notes

- In case **toUnit** is not specified, the system chooses a scale factor such that the integral is correct only if the flow is expressed in units per day. If the flow is expressed in units per hour, or per some other time unit, you must multiply this result by a conversion factor. The conversion factor equals the number of actual flow time units in a day.

  For instance, if you totalize values measured in gallons per minute, multiply the result of **TagTot** by 1440 to get the answer in gallons. This conversion factor is not related to the time period you are totalizing over; it is strictly a function of the attribute's engineering units.

Some PI System installations may have the default total period configured to be something other than per day, such as per hour. Your conversion factor would be different in this case. Users are recommended to verify the installation standard used at their site.

- Bad values are excluded from **TagTot** calculation.
- When the percentage of good data is less than 100%, **TagTot** determines the total based on good data and divides the fraction of good data in the interval. The time period during which the data is bad is ignored when calculating the total.

> **Caution:**
>
> If the attribute has very few good values during the time range, this function's result may not be trustworthy. Use the **PctGood** function to find out what percentage of the value is good.

Example

- `TagTot('att1', 't', '+1h')`

  [Return the totalized value of 'att1' between 12:00 and 1:00am today.]

- `TagTot('att1', 't', '+1h', "m")`

  [Return the totalized value of 'att1' calculated based on its unit between 12:00 and 1:00am today and convert to meters.]

- `TagTot('att1', 't', '+1h', 80)`

  [Return the totalized value of 'att1' between 12:00 and 1:00am today when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

- `TagTot('att1', 't', '+1h', "m", 80)`

  [Return the totalized value of 'att1' calculated based on its unit between 12:00 and 1:00am today and convert to meters when at least 80% of the values were good. Return error when minimum pctgood is not reached.]

TagType

For an attribute associated with a PI point, get a point's **point type** from the point database.

Syntax

```
TagType(attname)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

Point type

Exceptions

If the point does not exist, returns an error

Example

- `TagType('sinusoid')`

  [Returns **Float32**]

- `TagType('cdm158')`

  [Returns **Digital**]

TagTypVal

For an attribute associated with a PI point, get a point's typical value from the point database.

Syntax

```
TagTypVal(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's typical value. If the point is an integer or real number, a number is returned; if the point's value type is digital, this indicates a digital state

Exceptions

If the point does not exist, returns an error

Example

- `TagTypVal('sinusoid')`

  [Returns 50]

- `TagTypVal('cdm158')`

  [Returns 3, a digital state]

TagVal

Return the value of an attribute at a specified time.

Syntax

```
TagVal(attname [, t1])
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks

- **`t1`**

  Optional: A time expression. If you omit this argument, '*' is used.

Returns

The value of an attribute at a specified time. This value is interpolated unless the associated PI Point has the Step property set to `True` or the value type of the attribute cannot be interpolated (for example, if it is a string)

Exceptions

If the associated point does not exist or has no value at the specified time, an error is returned.

Example

- `TagVal('att1')`

  [Return the value of 'att1' at current time]

- `TagVal('att1', 't+8h')`

  [Return the value of 'att1' at 8am today]

- `TagVal('enum_att1')`

  [Return the value of 'enum_att1', an attribute from an enumeration set at current time]

TagZero

For an attribute with a PI point data reference, get a PI point's "zero" value from the point database.

Syntax

```
TagZero(attname)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotes

Returns

The point's zero value. If the point type is integer or real number, this is a number; if the point's type is digital, this is a digital state (character string)

Exceptions

If the point does not exist, returns an error

Example

- `TagZero('sinusoid')`

  [Returns 0]

- `TagZero('cdt158')`

  [Returns 50]

Tan

Return the trigonometric tangent of a number.

Syntax

```
Tan(x)
```

Arguments

- **x**

  Must be an integer or real number, which represents an angle in radians

Returns

The tangent of *x*.

Exceptions

If *x* is not a number, returns an error

Example

- `Tan(1)`

  [Returns 1.5574]

- `Tan(1.1)`

  [Returns 1.9648]

- `Tan('att1')`

  [Return the trigonometric tangent of the value of 'att1' at trigger time]

Tanh

Return the hyperbolic tangent of a number.

Syntax

```
Tanh(x)
```

Arguments

- **x**

  Must be an integer or real number

Returns

The hyperbolic tangent of *x*

Exceptions

If *x* is not a number, returns an error value

Example

- `Tanh(1)`

[Returns 0.76159]

- `Tanh(1.1)`

  [Returns 0.8005]

- `Tanh('att1')`

  [Return the hyperbolic tangent of the value of 'att1' at trigger time]

Text

Concatenate strings representing argument values.

Syntax

```
Text(x1 [, x2, … xn])
```

Arguments

- **`x1 … xn`**

  Any expression of normal PI value type

Returns

A string that is the concatenation of strings representing the argument values.

Example

- `Text(Round('sinusoid', 1), " is the current value of 'sinusoid' at ", '*')`

  [Returns rounded current value of 'sinusoid' at current time]

THEN

Operator that returns the first of two specified values when the conditional expression in IF-THEN-ELSE statement is True.

Syntax

```
IF(expression) THEN (x) ELSE (y)
```

Arguments

- **`expression`**

  Any expression that evaluates to true or false

- **`x , y`**

  An expression that evaluates to an output value

Returns

x when the conditional expression is true and *y* when the expression is false

Exceptions

None

Example

- `IF ('att1' > 50) THEN ('att2') ELSE ('att3')`

  [If the value of 'att1' is greater than 50, then return the value of 'att2'; otherwise return the value of 'att3']

TimeEq

Find the total time, within a range, when an attribute value is equal to a specified value. Time returned is in seconds.

Syntax

```
TimeEq(attname, starttime, endtime, x)
```

Arguments

- **`attname`**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **`starttime`**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **`endtime`**

- **`endtime`**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **`starttime`**

- **`x`**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is equal to a given value.

Notes

Bad values are excluded from **`TimeEq`** calculation.

Example

- `TimeEq('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was equal to 80.]

- `TimeEq('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was equal to the value at 8am today. Return the result in seconds.]

- `TimeEq('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeGE

Find the total time, within a range, when an attribute value is greater than or equal to a specified value. Time returned is in seconds.

Syntax

```
TimeGE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is greater than or equal to a given value.

Exceptions

None

Notes

TimeGE interpolates between events, if necessary, to find the times when the attribute value crossed the given value.

Bad values are excluded from **TimeGE** calculation.

Example

- `TimeGE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was greater than or equal to 80.]

- `TimeGE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was greater than or equal to the value at 8am today. Result is in seconds.]

- `TimeGE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeGT

Find the total time, within a range, when an attribute value is greater than a specified value. Time returned is in seconds.

Syntax

```
TimeGT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is greater than a given value.

Exceptions

None

Notes

TimeGT interpolates between events, if necessary, to find the times when the attribute value crossed the specified value.

Bad values are excluded from **TimeGT** calculation.

Example

- `TimeGT('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was greater than 80.]

- `TimeGT('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was greater than the value at 8am today. Result is in seconds.]

- `TimeGT('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeLE

Find the total time, within a range, when an attribute value is less than or equal to a given value. Time returned is in seconds.

Syntax

```
TimeLE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is less than or equal to a given value.

Exceptions

None

Notes

TimeLE interpolates between archive events, if necessary, to find the times when the point crossed the given value.

Bad values are excluded from **TimeLE** calculation.

Examples

- `TimeLE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was less than or equal to 80.]

- `TimeLE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was less than or equal its value at 8am today. Result is in seconds.]

- `TimeLE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeLT

Find the total time, within a range, when an attribute value is less than a specified value. Time returned is in seconds.

Syntax

```
TimeLT(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is less than a given value.

Exceptions

None

Notes

TimeLT interpolates between Archive events, if necessary, to find the times when the attribute value crossed the specified value.

Bad values are excluded from **TimeLT** calculation.

Example

- `TimeLT('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was less than 80.]

- `TimeLT('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was less than its value at 8am today. Result is in seconds.]

- `TimeLT('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeNE

Find the total time, within a range, when an attribute value is not equal to a specified value. Time returned is in seconds.

Syntax

```
TimeNE(attname, starttime, endtime, x)
```

Arguments

- **attname**

  The name of an attribute with a PI point data reference, enclosed in single quotation marks.

- **starttime**

  time expression representing the beginning of a time range enclosed in single quotes; can be a relative time (such as `'-3h'`) in reference to an absolute **endtime**

- **endtime**

  time expression representing the end of a time range enclosed in single quotes; can be a relative time (such as `'+1h'`) in reference to an absolute **starttime**

- **x**

  The reference value of the search; must be an enumeration set (string), integer or real number

Returns

The total time (in seconds), within a range, when an attribute value is not equal to a specified value.

Exceptions

None

Notes

Bad values are excluded from **TimeNE** calculation.

Example

- `TimeNE('att1', 't', '+1h', 80)`

  [Find the total time between 12:00 and 1:00am today when 'att1' was not equal to 80.]

- `TimeNE('att1', '-1h', '*', TagVal('att1', 't+8h'))`

  [Find the total time in the past hour when the value of 'att1' was not equal to the value at 8am today. Result is in seconds.]

- `TimeNE('enum_att1', '*-1d', '*', DigState("Normal", 'enum_att1'))`

TimeStamp

Return the time stamp for a single time-stamped value or an array of time-stamped values.

Syntax

```
TimeStamp(x)
```

Arguments

- **x**

  A time-stamped value or an array of time-stamped values

Returns

The timestamp of the specified time-stamped value

Exceptions

If the value of *x* is `No Data`, then `Calc Failed` is returned

Example

- `TimeStamp(PrevVal('sinusoid', '*'))`

  [Return the time stamp of the last recorded value before the current one for 'sinusoid']

- `TimeStamp(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5]
Variable 2	TimeStamp(Variable1)	[1/3/2018 8:15:00 AM, 1/3/ 8:16:17 AM, 1/3/2018 8:20:00 AM]

  [Assuming current time is 8:23 AM on 1/3/2018, returns the timestamps of 3 recorded values in the past 10 minutes]

Total

Return the sum of two or more values.

Syntax

```
Total(x1, x2 [, ... xn])
```

Arguments

- **x1, ... xn**

  Numbers, time intervals or a single array of numbers or time intervals

Returns

The total of the arguments. The result has the same data type as the arguments.

Exceptions

Arguments whose run-time values are digital states are not included in the total. If all values are digital states, **`Total`** returns an error value. Returns an error if the array does not consist of same value type.

Notes

Unit of Measure (UOM) of the arguments is carried over to the result when at least one argument has a defined UOM while others don't. The returned value lacks a UOM if arguments have different UOMs.

Example

- `Total('att1', 'att2', TagVal('att1', 'y+2h'), 40)`

  [Return the sum of following values: values of 'att1' and 'att2' at current time, the value of 'att1' at 2am yesterday and 40]

- `Total('t'-'y', '+1h')`

  [Returns 1.01:00:00]

- `Total(Variable1)`

  Note:

Name	Expression	Value at Evaluation
Variable1	RecordedValues('att1', '*-10m', '*')	[1, 3, 5, 7, 9]
Variable 2	Total(Variable1)	25

  [Return the sum of values in an array `Variable1`]

Trim

Trim blanks on both sides of a string.

Syntax

```
Trim(s1)
```

Arguments

- **`s1`**

  string

Returns

The source string with leading and trailing blanks removed.

Exceptions

If *s1* is not a string, an error value is returned.

Example

- `Trim(" String ")`

[Returns "String"]

- `Trim(" String is a string attribute. ")`

  [Returns "String is a string attribute."]

Trunc

Truncate a number or time to the next lower unit.

Syntax

```
Trunc(x [, unit])
```

Arguments

- **x**

  An integer or real number, time expression, or time period.

- **unit**

  Optional. The size of the unit to truncate to; *x* will be truncated to a multiple of *unit*. If *x* is a number, *unit* must be a number. If *x* is a time expression or time period, *unit* must be a time period. If *unit* is omitted, **Trunc** truncates to the next lower integer (for a number) or second (for a time period).

Returns

The largest multiple of unit that is less than *x*. For a negative *x*, it returns the lowest multiple of *unit* larger than *x*. The return is the same data type as *x*.

Exceptions

If *x* is a string, or if unit is of the wrong data type, an error is returned.

Notes

If *x* is a time, and unit is omitted, this routine has no effect, as times are only accurate to one second.

When |x| < |unit|, 0 is returned.

Example

- `Trunc(12.999)`

  [Returns 12, truncated to the next lower integer]

- `Trunc(28.75, 10)`

  [Returns 20, truncated to next lower multiple of 10]

- `Trunc('14-Dec-16 11:47', '+1h')`

  [Returns 12/14/2016 11:00:00 AM, truncated to next lower hour]

- `Trunc('18:47'-'15:00','+1h')`

  [Returns 03:00:00, truncated period to next lower hour]

> **Note:**
> Truncating to the next lower day results in a timestamp of the next lower day in UTC time, not local time.

UCase

Convert a string to an uppercase string.

Syntax

```
UCase(s1)
```

Arguments

- **s1**

  String in double quotes

Returns

s1 in uppercase

Exceptions

If the argument is not a string, returns an error value.

Example

- `UCase("String")`

  [Returns "STRING"]

Weekday

Extract the day of the week from a given time expression.

Syntax

```
Weekday(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The day of the week, in the range 1-7, where 1 represents a Sunday.

Exceptions

None

Example

- `Weekday('*')`

[Return what day of the week today is]

- `Weekday(FindEq('att1', '-7d', '*', 50))`

  [Return what day of the week it was when the value of 'att1' was 50 for the first time in the past 7 days]

Year

Extract the year from a time expression.

Syntax

```
Year(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The year of time, in the range 1970-present

Exceptions

None

Example

- `Year('*')`

  [Return what year it is now]

- `Year(FindGT('att1', '1/1/1970', '*', 50))`

  [Return what year it was when the value of 'att1' was first greater than 50 since 1/1/1970]

Yearday

Extract the day of the year from a time expression. The day of the year (also known as a Julian day) is an integer ranging from 1 to 366, where 1 represents January 1.

Syntax

```
Yearday(t1)
```

Arguments

- **t1**

  A time expression, enclosed in single quotes

Returns

The day of the year of time, in the range 1-366, where 1 represents January 1

Exceptions

None

Example

- `Yearday('*')`

  [Return what day of the year today is, where 1 represents January 1]

- `Yearday('y')`

  [Return what day of the year yesterday was, where 1 represents January 1]

Steam functions for analysis expressions

You can include steam functions in analysis expressions to calculate the thermodynamic properties of steam. These functions are based on the 1997 IAPWS Industrial Formulation. The implementation is based on the combination of officially published forward and backward equations for all applicable regions.

> **Note:**
> Avoid confusing these steam functions with a similar set of steam functions used for PI Performance Equations in tag-based analytics. For more information on steam functions in asset and tag-based analytics, see Steam functions in asset and tag-based analytics.

The table below lists supported functions and descriptions. Ranges and units of measure for steam function properties are detailed in Ranges for steam function inputs. Reference states that apply to steam are listed in Steam property reference states.

Steam function descriptions

Function	Description
Steam_TsatP	Saturation temperature as a function of pressure.
Steam_HsatP	Saturation vapor specific enthalpy as a function of pressure.
Steam_SsatP	Saturation vapor specific entropy as a function of pressure.
Steam_VsatP	Saturation vapor specific volume as a function of pressure.
Steam_PsatT	Saturation pressure as a function of temperature.
Steam_HsatT	Saturation vapor specific enthalpy as a function of temperature.
Steam_SsatT	Saturation vapor specific entropy as a function of temperature.
Steam_VsatT	Saturation vapor specific volume as a function of temperature.
Steam_VPT	Vapor specific volume as a function of pressure and temperature.
Steam_VPTL	Liquid specific volume as a function of pressure and temperature. (For water.)
Steam_VPH	Vapor specific volume as a function of pressure and enthalpy.
Steam_VPS	Vapor specific volume as a function of pressure and entropy.
Steam_HPT	Vapor specific enthalpy as a function of pressure and temperature.
Steam_HPTL	Liquid specific enthalpy as a function of pressure and temperature. (For water.)
Steam_HPS	Vapor specific enthalpy as a function of pressure and entropy.
Steam_SPT	Vapor specific entropy as a function of pressure and temperature.
Steam_SPTL	Liquid specific entropy as a function of pressure and temperature. (For water.)

Function	Description
Steam_SPH	Vapor specific entropy as a function of pressure and enthalpy.
Steam_TPH	Temperature as a function of enthalpy and pressure.
Steam_TPS	Temperature as a function of entropy and pressure.
Steam_XPH	Steam quality (vapor fraction) as a function of pressure and enthalpy. (For wet steam.)
Steam_XPS	Steam quality (vapor fraction) as a function of entropy and pressure. (For wet steam.)
Steam_VPX	Steam specific volume as a function of pressure and steam quality. (For wet steam.)
Steam_HPX	Specific enthalpy as a function of pressure and steam quality. (For wet steam.)
Steam_SPX	Steam specific entropy as a function of pressure and steam quality. (For wet steam.)

Topics in this section

- Steam functions in asset and tag-based analytics
- Ranges for steam function inputs
- Unit of measure conversion for steam functions
- Steam property reference states
- Steam functions reference

Steam functions in asset and tag-based analytics

Asset and tag-based analytics tools both include steam functions that calculate thermodynamic properties of steam and water. Both sets of tools provide a complete set of steam functions but are based on different standards and have different implementations.

- **Steam functions in asset analytics**

  Steam functions in asset analytics are based on the 1997 IAPWS Industrial Formulation. The implementation is based on the combination of officially published forward and backward equations for all applicable regions.

  Note:
  The steam function names in asset analytics begin with "Steam_".

- **Steam functions in tag-based analytics**

  Steam functions in tag-based analytics refer to the PI PE Steam Functions Module (PI Steam), which is an extension to the PI Performance Equations Scheduler. These functions are based on the ASME Steam Tables, 6th Ed., and are also available in a COM library, `PISteamTableFunctions.dll`, which is distributed by other OSIsoft products, such as PI ACE.

  Note:
  Tag-based steam functions support both English and SI units; the function names begin with "StmSI_" or "StmEng_" for SI units and English units respectively.

Ranges for steam function inputs

The table below shows ranges and default units of measure for each of the five properties involved in steam function calculations.

The default units of measure are SI units. For information about using steam functions with other units of measure, see Unit of measure conversion for steam functions.

Pressure affects the state of steam. Several things to keep in mind for functions with pressure as an input are described in Considerations for steam functions that have pressure as an input.

Range of inputs for steam function properties

Function	Temp (°C)	Pressure (kPa)	Enthalpy(J/g)	Entropy(J/(g K))	Quality
Steam_TsatP		0.611212678 to 22064			
Steam_HsatP		0.611212678 to 22064			
Steam_SsatP		0.611212678 to 22064			
Steam_VsatP		0.611212678 to 22064			
Steam_PsatT	0.0 to 373.946				
Steam_HsatT	0.0 to 373.946				
Steam_SsatT	0.0 to 373.946				
Steam_VsatT	0.0 to 373.946				
Steam_VPT	0.0 to 800	0.611212678 to 100000			
Steam_VPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_VPTL	0.0 to 800	0.611212678 to 100000			
Steam_VPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_VPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_HPT	0.0 to 800	0.611212678 to 100000			
Steam_HPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_HPTL	0.0 to 800	0.611212678 to 100000			
Steam_HPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_SPT	0.0 to 800	0.611212678 to 100000			

Function	Temp (°C)	Pressure (kPa)	Enthalpy(J/g)	Entropy(J/(g K))	Quality
Steam_SPT (high temperature and pressure)	800 to 2000	0.611212678 to 50000			
Steam_SPTL	0.0 to 800	0.611212678 to 100000			
Steam_SPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_TPH		0.611212678 to 100000	-0.04159 to 4160.7		
Steam_TPS		0.611212678 to 100000		-0.000155 to 11.921	
Steam_XPH		0.611212678 to 22064	-0.04159 to 4160.7		
Steam_XPS		0.611212678 to 22064		-0.000155 to 11.921	
Steam_VPX		0.611212678 to 22064			0.0 to 1.0
Steam_HPX		0.611212678 to 22064			0.0 to 1.0
Steam_SPX		0.611212678 to 22064			0.0 to 1.0

Considerations for steam functions that have pressure as an input

Because pressure affects the state of steam, be aware of these considerations for functions that have pressure as an input.

Functions with pressure and temperature as inputs

Steam_HPT, **Steam_SPT** and **Steam_VPT** can be used for compressed water, superheated or saturated dry steam. For input temperature and pressure on the saturated curve, the calculated entropy, enthalpy or volume is a property of saturated vapor. These three functions can also accept high-range input temperatures (800 to 2000 °C) with pressures from 0.611212678 to 50000 kPa.

Steam_HPTL, **Steam_SPTL** and **Steam_VPTL** are used only for compressed water.

Functions with pressure and enthalpy or entropy as inputs

Steam_VPH, **Steam_VPS**, **Steam_HPS**, **Steam_SPH**, **Steam_TPH** and **Steam_TPS** can be used for compressed water, superheated or wet steam. For these functions, the range for enthalpy (H) is -0.04159 to 4160.7 J/g and for entropy (S) is -0.000155 to 11.921 J/(g K).

For **Steam_XPH** and **Steam_XPS**, input enthalpy or entropy greater than saturated vapor properties or less than saturated liquid properties returns an error.

Unit of measure conversion for steam functions

The default units of measure (UOM) for the steam functions are SI units. Any inputs to the steam functions defined in English units are automatically converted to the corresponding SI units. Similarly, if a steam function is configured with an output attribute, its output is automatically converted to the UOM configured for that attribute.

> **Note:**
> Automatic conversion requires that all UOM have been defined properly in the UOM database. If the definition of conversion from one unit to another of the same type is missing from the UOM database, conversion will fail.

In the expression editor, when you evaluate a steam function expression, results are displayed in the default SI units. To display the results in other UOM, you can use the **`Convert`** function.

The following example expressions use numeric values, variables, and attributes to show how steam functions handle UOM for input and output quantities.

Numeric input values

Numeric input values are recognized as quantities in the default UOM. In the following example for **`Steam_VPT`**, the numeric inputs are recognized as 300 kPa and 200 °C, and the output value is in cm^3/g, the default UOM for specific volume.

Name	Expression	Value	Output Attribute
`Variable1`	`Steam_VPT(300, 200)`	`716.44 cm3/g`	`Click to map`

Input and output attributes

In the following expression, **`Steam_VsatT`** takes as its only input the attribute 'TempSat', whose current value is 300 °F. The temperature is automatically converted to °C to evaluate the expression, and the result is displayed in the default SI UOM for **`Steam_VsatT`**.

Name	Expression	Value	Output Attribute
`Variable1`	`Steam_VsatT('TempSat')`	`403.7 cm3/g`	`OutputVolume`

The result is also mapped to the 'OutputVolume' output attribute. This attribute is configured in English UOM (ft^3/lb), so the result is automatically converted to that UOM. (On the Attributes tab in PI System Explorer, 'OutputVolume' would be listed with a value of 6.466627 ft^3/lb.)

Variables and the Convert function

This example uses variables and the **`Convert`** function to specify quantities in English UOM as steam function inputs.

1. In the P and T rows of the expression below, **`Convert`** assigns numeric values to English UOM for pressure and temperature.
2. In the B row, the variables P and T are automatically converted to SI units for the **`Steam_VPT`** calculation; the result is displayed in default SI UOM. The result is also mapped

to the output attribute 'OutputVolume'. That attribute is configured in English UOM (ft^3/lb), so the result is automatically converted to that UOM.

3. In the last row, **Convert** is used to convert the result (B) to English UOM and display it in the expression.

Name	Expression	Value	Output Attribute
P	Convert(2000, "psia")	2000 psia	Click to map
T	Convert(800, "°F")	800 °F	Click to map
B	Steam_VPT(P, T)	19.206 cm3/g	OutputVolume
Volume	Convert(B, "ft3/lb")	0.30762 ft3/lb	DisplayedVolume

Steam property reference states

The American Society of Mechanical Engineers (ASME) establishes the following reference states:

- **triple point**

  Triple point of water is at 273.16 K or 0.01 °C or 32.018 °F.

- **Celsius scale**

  The Celsius temperature is exactly T_k - 273.15.

- **critical point**

  Critical point of steam is at 647.096 K and 22,064 kPa, or 705.103 °F and 3200.1 psia.

- **reference state**

  The specific internal energy and specific entropy of the liquid phase were fixed at zero at the triple point of water.

Steam functions reference

Steam functions available for use in expression analyses are listed alphabetically.

Topics in this section

- Steam_HPS
  Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_HPT
  Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_HPTL
  Calculate the liquid specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_HPX
  Calculate the specific enthalpy as a function of pressure (**P** in kPa) and quality (vapor fraction).

- Steam_HsatP
  Calculate the saturated vapor specific enthalpy as a function of pressure (**P** in kPa).
- Steam_HsatT
  Calculate the saturated vapor specific enthalpy as a function of temperature (**T** in °C).
- Steam_PsatT
  Calculate the saturation pressure as a function of temperature (**T** in °C).
- Steam_SPH
  Calculate the vapor-specific entropy as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_SPT
  Calculate the vapor specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_SPTL
  Calculate the liquid specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_SPX
  Calculate the steam-specific entropy as a function of pressure (**P** in kPa) and quality (vapor fraction).
- Steam_SsatP
  Calculate the saturated vapor-specific entropy as a function of pressure (**P** in kPa).
- Steam_SsatT
  Calculate the saturated vapor-specific entropy as a function of temperature (**T** in °C).
- Steam_TPH
  Calculate the steam temperature as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_TPS
  Calculate the steam temperature as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_TsatP
  Calculate the saturation temperature as a function of pressure (**P** in kPa).
- Steam_VPH
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and enthalpy (**S** in J/g).
- Steam_VPS
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).
- Steam_VPT
  Calculate the vapor-specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_VPTL
  Calculate the liquid-specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C).
- Steam_VPX
  Calculate the steam-specific volume as a function of pressure (**P** in kPa) and quality (vapor fraction).

- Steam_VsatP
  Calculate the saturated vapor-specific volume as a function of pressure (**P** in kPa).
- Steam_VsatT
  Calculate the saturated vapor-specific volume as a function of temperature (**T** in °C).
- Steam_XPH
  Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g).
- Steam_XPS
  Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)).

Steam_HPS

Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed vapor specific enthalpy in J/g.

Example

- `Steam_HPS(2500, 7.5956)`

  [Returns 3684.9 J/g]

Steam_HPT

Calculate the vapor specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific enthalpy in J/g.

Example

- `Steam_HPT(40000, 600)`

  [Returns 3350.4 J/g]

Steam_HPTL

Calculate the liquid specific enthalpy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Only use for compressed water. For any **T** greater than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific enthalpy in J/g.

Example

- `Steam_HPTL(40000, 100)`

  [Returns 449.27 J/g]

Steam_HPX

Calculate the specific enthalpy as a function of pressure (**P** in kPa) and quality (vapor fraction). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed specific enthalpy in J/g.

Example

- `Steam_HPX(10000, 0.9)`

  [Returns 2593.7 J/g]

Steam_HsatP

Calculate the saturated vapor specific enthalpy as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed specific enthalpy for saturated vapor in J/g.

Example

- `Steam_HsatP(10000)`

  [Returns 2725.5 J/g]

Steam_HsatT

Calculate the saturated vapor specific enthalpy as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_HsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific enthalpy in J/g.

Example

- `Steam_HsatT(250)`

  [Returns 2801 J/g]

Steam_PsatT

Calculate the saturation pressure as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_PsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturation pressure of the steam in kPa.

Example

- `Steam_PsatT(250)`

  [Returns 3975.9 kPa]

Steam_SPH

Calculate the vapor-specific entropy as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed vapor specific entropy in J/(g K).

Example

- `Steam_SPH(10000, 3500)`

  [Returns 6.756 J/(g K)]

Steam_SPT

Calculate the vapor specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific entropy in J/(g K).

Example

- `Steam_SPT(40000, 600)`

  [Returns 6.017 J/(g K)]

Steam_SPTL

Calculate the liquid specific entropy as a function of pressure (**P** in kPa) and temperature (**T** in °C). Only use for compressed water. For any **T** higher than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific entropy in J/(g K).

Example

- `Steam_SPTL(30000, 500)`

  [Returns 5.7956 J/(g K)]

Steam_SPX

Calculate the steam specific entropy as a function of pressure (**P** in kPa) and quality (vapor fraction). Use for only wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed specific entropy of the steam in J/(g K)

Example

- `Steam_SPX(15000, 0.7)`

  [Returns 4.8229 J/(g K)]

Steam_SsatP

Calculate the saturated vapor specific entropy as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed saturated vapor specific entropy in J/(g K)

Example

- `Steam_SsatP(10000)`

  [Returns 5.6159 J/(g K)]

Steam_SsatT

Calculate the saturated vapor specific entropy as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_SsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific entropy in J/(g K).

Example

- `Steam_SsatT(250)`

  [Returns 6.0722 J/(g K)]

Steam_TPH

Calculate the steam temperature as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed steam temperature in °C.

Example

- `Steam_TPH(40000, 3500)`

  [Returns 643.48 °C]

Steam_TPS

Calculate the steam temperature as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed steam temperature in °C.

Example

- `Steam_TPS(40000, 6)`

  [Returns 595.93 °C]

Steam_TsatP

Calculate the saturation temperature as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_TsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed saturation temperature in °C.

Example

- `Steam_TsatP(10000)`

  [Returns 311 °C]

Steam_VPH

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and enthalpy (**S** in J/g). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed vapor specific volume in cm^3/g.

Example

- `Steam_VPH(25000, 3500)`

  [Returns 14.197 cm^3/g]

Steam_VPS

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use for both superheated and wet steam as well as compressed water. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed vapor specific volume in cm^3/g

Example

- `Steam_VPS(40000, 6)`

  [Returns 8.0055 cm^3/g]

Steam_VPT

Calculate the vapor specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C). Use for an entire range from compressed water to superheated steam. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPT(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed vapor specific volume in cm^3/g.

Example

- `Steam_VPT(50000, 500)`

  [Returns 3.8894 cm^3/g]

Steam_VPTL

Calculate the liquid specific volume as a function of pressure (**P** in kPa) and temperature (**T** in °C). For any **T** higher than the saturation temperature for **P**, the function returns an out-of-range error. The function accepts any **T** in the temperature range as long as **P** is greater than the critical pressure. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPTL(P, T)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 100000 kPa .

- **T**

  Temperature of the steam. **T** can range from 0.0 to 800 °C.

Returns

Computed liquid specific volume in cm^3/g

Example

- `Steam_VPTL(75000, 500)`

  [Returns 2.308 cm^3/g]

Steam_VPX

Calculate the steam specific volume as a function of pressure (**P** in kPa) and quality (vapor fraction). Use for only wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VPX(P, X)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **X**

  Steam quality (vapor fraction). **X** can range from 0.0 to 1.0.

Returns

Computed steam specific volume in cm^3/g

Example

- `Steam_VPX(15000, 0.7)`

  [Returns 7.7352cm^3/g]

Steam_VsatP

Calculate the saturated vapor specific volume as a function of pressure (**P** in kPa). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VsatP(P)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

Returns

Computed vapor specific volume in cm^3/g

Example

- `Steam_VsatP(10000)`

  [Returns 18.034 cm^3/g]

Steam_VsatT

Calculate the saturated vapor specific volume as a function of temperature (**T** in °C). By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_VsatT(T)
```

Arguments

- **T**

  Temperature of the steam. **T** can range from 0.0 to 373.946 °C.

Returns

Computed saturated vapor specific volume in cm^3/g.

Example

- `Steam_VsatT(250)`

  [Returns 50.087 cm^3/g]

Steam_XPH

Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and enthalpy (**H** in J/g). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_XPH(P, H)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **H**

  Enthalpy of the steam. **H** can range from -0.04159 to 4160.7 J/g.

Returns

Computed steam quality (vapor fraction)

Example

- `Steam_XPH(10000, 2500)`

  [Returns 0.82888]

Steam_XPS

Calculate the steam quality (vapor fraction) as a function of pressure (**P** in kPa) and entropy (**S** in J/(g K)). Use only for wet steam. By default, input arguments and returned values are in SI units. You can change the units of measure, for example, by using the `Convert` function.

Syntax

```
Steam_XPS(P, S)
```

Arguments

- **P**

  Pressure of the steam. **P** can range from 0.611212678 to 22064 kPa.

- **S**

  Specific entropy of the steam. **S** can range from -0.000155 to 11.921J/(g K).

Returns

Computed steam quality (vapor fraction)

Example

- `Steam_XPS(10000, 5)`

  [Returns 0.72695]

Asset analytics native integration of external systems

MATLAB® Production Server™ is the first external system that asset analytics is integrating. By interfacing with the external system, users will be able to perform advanced calculations in expression and event frame generation analyses using the extended library of functions that reside outside of asset analytics.

Topics in this section

- Integration of MATLAB Production Server into asset analytics
- Configure an external system endpoint
- Supported functions and data types in MATLAB Production Server
- Error handling

Integration of MATLAB Production Server into asset analytics

Introduction

Beginning with the 2018 release, users of asset analytics can enjoy the extended capabilities of customized functions from MATLAB® Production Server™. Those who wish to make use of this functionality must procure and install MATLAB Production Server.

Requirements

Below is a simplified diagram depicting the connection and data transfer between asset analytics and MATLAB Production Server.

In order to take advantage of the extended features and functionality, users will need the following:

- PI Asset Framework 2018
- MATLAB Production Server 2018a with service discovery turned on. See the service discovery section in Supported functions and data types in MATLAB Production Server.
- MATLAB, MATLAB Compiler and MATLAB Compiler SDK
- Valid certificate if using the HTTPS schema for connection

Elements in the dotted box above must be set up before you can interface with the system at OSIsoft. For more information, refer to the MATLAB Production Server documentation (https://www.mathworks.com/help/mps/index.html).

Configure an external system endpoint

To configure MATLAB Production Server as an endpoint, see Configure an external system endpoint.

What to expect upon configuration

Once properly configured, you will be able to see all the available functions in the MATLAB Production Server shown within the functions list located on the right hand side of the Analyses tab. Expand **Functions** and select **MATLAB** from the drop-down list. Same as other asset analytics expression functions, function name, description, number of input and output arguments as well as value type will be displayed on the lower right-hand corner. You can click **Reload** to get the latest functions if you have made changes to them. Note that the automatic refresh occurs in the background periodically and this button is more for an immediate effect. If any functions found in the MPS are not supported, they will be greyed out and a tooltip will provide additional information.

Performance considerations

If you queue a lot of expensive high frequency calculations, it may negatively impact the overall performance of your system. We recommend that you use discretion and allocate resources accordingly. Avoiding frequent calls to the external system is also encouraged.

Configure an external system endpoint

Before you start

You must have the PI System Explorer Management plug-in installed before you can configure the external system endpoint. Make sure that you have a valid certificate if you are using the HTTPS scheme. Both HTTP and HTTPS are supported.

Procedure

1. Open PI System Explorer.
2. Go to **Tools** and select **External Systems**. Manage External Systems opens.
3. Enter the **Host** and **Port** information and click **Test Connection**. Make sure that you include a full host name including HTTP or HTTPS prefix. Upon successful connection, a list of available packages and the number of functions will be displayed.
4. Click **Save**.

Supported functions and data types in MATLAB Production Server

Supported functions

We only support functions with a proper function signature file. For each input and output argument, the type must be specified. See below for supported and unsupported data types.

Supported data types

The table here lists the MATLAB data types that asset analytics support:

MATLAB Data Type	Comment
Numeric (double, single, int8, uint8, int16, uint16, int32, uint32, int64, uint64)	Single value or one-dimensional array only
Char	
Logical (True, False)	
Cell	Single value or one-dimensional array only. As the string array is not a defined data type in MATLAB Production Server, we recommend using the cell array as an alternative.

Note:
If not specified in the function argument description, arrays are sent as row vectors from PI AF to the MATLAB Production Server.

Asset analytics will make best effort to convert the MATLAB Production Server data type into expected PI AF attribute type. In case the conversion fails, the function call will return a failure message.

With MATLAB Production Server 2018a, it is not yet possible to pass the value and the timestamp as a single argument from asset analytics. Therefore, this information needs to be sent on as two separate arguments to the MATLAB Production Server. In addition, the timestamp must be declared as data type double within the MATLAB code because the timestamp would be passed as POSIX time (also known as UNIX epoch time) when calling the MATLAB function.

In the following example, an array of values are retrieved from 'attribute1' within the specified time period by way of an expression function **`RecordedValues`** in asset analytics. Then the values and the timestamps are passed on to a fictitious function called MATLAB:CurveFitting.PolynomialFit, where MATLAB is the external system, CurveFitting is the name of the package and PolynomialFit is the function name.

Note:

Name	Expression
Values	RecordedValues('attribute1', 't', '*')
Variable1	MATLAB:CurveFitting.PolynomialFit(Values, TimeStamp(Values))

Within the MATLAB code, users must include a way to handle the timestamps that are passed on from asset analytics as POSIX time and convert it into a form that is consumable by MATLAB's native datetime object. This can be done in the following way, for example:

```
function calculated = PolynomialFit(values, timestamps)
timestampsConverted = datetime(timestamps,'ConvertFrom','posixtime')
```

For more information, see MATLAB documentation on datetime (https://www.mathworks.com/help/matlab/ref/datetime.html).

Note:
Time intervals in expression analyses performance equation syntax will be converted to seconds.

Data types not supported

Multi-dimensional array, struct and complex numbers are the MATLAB Production Server data types not supported in asset analytics.

Service discovery

The signature file in MATLAB Production Server is per package. The completeness of the signature is checked per function. This file must be added and/or modified each time a function is deployed.

Error handling

You will see relevant error messages surface in the UI to guide you.

There are 2 main types of errors: fatal and runtime. There are other errors not listed here.

- **Fatal errors**

  Analysis goes into an error as a result

  - Incorrect namespace of the MATLAB function
  - Incorrect package or a function name

  - Unsupported or incorrect number of arguments in a function
  - Invalid certificate if using HTTPS scheme

- **Runtime errors**

  Analysis does not go into error and the function evaluates to **Calc Failed**. Examples below are some common errors:

  - Divided by zero
  - Not a number when a number is required as an argument or as a MATLAB function output
  - Error in the MATLAB code; the error message will expose the type and location of the error
  - Connection is unavailable: server not available, expired MATLAB license and/or other HTTP/HTTPS errors
  - MATLAB function evaluation timeout: current setting is after 10 minutes

PI Analysis Service management

(These options are available if your installation of PI System Explorer includes the **Management** plug-in and you are signed in to a user account with permissions to configure PI Analysis Service.)

PI Analysis Service runs all analyses in the current PI AF server. Two features help you manage the service:

- The Analysis Service Statistics window enables you to explore statistics about the service to monitor its operations. These statistics are particularly important in diagnosing and identifying a solution for performance issues. You can save these statistics as a text file, which can provide a detailed snapshot of your analysis service to share with support staff helping you troubleshoot a problem. See View analysis service statistics.
- The Configuration window enables you to view or modify settings for analysis service properties to adjust performance. See View or modify analysis service.

View analysis service statistics

Procedure

1. In the navigator, click **Management**, and then select **Analysis** to view the analysis management area.
2. Right-click anywhere in the **Operations** panel and click **View Analysis Service Statistics**. The Analysis Service Statistics window displays groups of statistics you can use to monitor analysis service operations or to diagnose a performance issue and identify a solution.
3. Optional. Click **Refresh** to update statistics with current data.
4. Optional. Click **Save** to save current statistics as a text file. This enables you to share a snapshot of the current state of your analysis service with customer support staff.

View or modify analysis service

> **Note:**
> Changing these settings can greatly impact system performance. If you have any questions about changing the default settings, contact customer support staff for help.

Procedure

1. In the navigator, click **Management**, and then select **Analysis** to view the analysis management area.
2. Right-click anywhere in the **Operations** panel, and then click **Edit Analysis Service Configuration**.

   The Configuration window lists settings for analysis service properties. See Analysis service configuration for details about these settings.
3. To modify a property, edit its setting in the **Configuration** column.
4. Click **Ok** to apply any changes and close the window.

Analysis service configuration

You can modify these settings in the Configuration window. For information on how to configure the parameters below, see View or modify analysis service.

> **Note:**
> Changes to the **`AutoBackfillingEnabled`**, **`MaximumAllowedAutoBackfillingSpanInHours`** or **`NumberParallelDataPipes`** settings require a restart of PI Analysis Service to take effect.

- **AutoBackfillingEnabled**

  This property enables or disables automatic backfilling of gaps in data that result from periods when PI Analysis Service is not active. Automatic backfilling is enabled by default. In some cases, it may not be needed (in a test environment, for example). Enter `False` to disable this property. If you update this setting, you must restart PI Analysis Service.

  When an analysis is resumed after a short downtime, real-time calculations are started and, concurrently, auto-backfilling is also queued, if enabled. However, events with time-stamps before service start-time are processed differently depending on whether auto-backfilling is enabled using the **`AutoBackfillingEnabled`** configuration setting:

  - If auto-backfilling is enabled, triggered events with time-stamps before service start-time are processed by auto-backfilling and not by real-time processing.
  - If auto-backfilling is disabled, real-time processing will process all triggered events that are retrieved after the analysis service is started, including events that have time-stamps before analysis start-time.

- **AutoRecalculationEnabled**

  This property enables or disables automatic recalculation of analyses. Automatic recalculation executes analyses when a new input with a time stamp before the latest evaluated trigger is received. This useful feature automatically recalculates late-arriving inputs. Automatic recalculation is enabled by default. Enter `False` to disable this property.

> **Note:**
> Although automatic recalculation is enabled globally by setting this parameter, you need to opt in for an automatic recalculation at individual analysis (template) level. For more information, see Configure automatic recalculation of analyses and analyses templates.

- **AutoRecalculationIgnoreTimeInSeconds**

  This setting allows users to specify a duration whereby any input events with time stamps within this duration and older than the latest trigger time are ignored for automatic recalculation. The default value for this property is 30 seconds. Setting this parameter can be useful for filtering out noises in some analysis inputs that are not desirable for automatic recalculation, for example, if they are known to be delayed.

- **AutoRecalculationMinWaitTimeInSeconds**

  This setting controls how long analysis service will wait before queuing an analysis for automatic recalculation upon receiving an out-of-order triggering event. A valid range for this setting is 30 to 900 seconds, with a default of 60 seconds. The wait time is reset with the arrival of new out-of-order triggering events. Regardless of this setting, any analyses that have had an out-of-order triggering event will be queued for recalculation after 900 seconds.

- **CacheTimeSpanInMinutes**

  PI Analysis Service dynamically determines data cache time span for AF attributes used as inputs in calculations. For example, for analyses calculating hourly summary, PI Analysis Service would automatically cache an hour worth of data for required input attributes. This ensures that data required for calculations is available in the data cache, while unnecessary data can be trimmed. Manually set **`CacheTimeSpanInMinutes`** only if the time span cannot be determined by calculation configuration. A longer time setting means that any input data needed for calculations is available in the data cache, but it uses more system resources. For calculations that require earlier data (for example, from an hour earlier), you can specify a longer period for this setting to keep data cached, thus avoiding calls to PI Data Archive. The maximum time span that would be cached is 7 days and any data in the time span beyond 7 days would be trimmed. Note that the **`MaxCacheEventsPerAttribute`** setting takes precedence over this setting, so both settings may have to be adjusted simultaneously.

- **CalculationWaitTimeInSeconds**

  A period of time to wait before evaluating scheduled analyses. This wait time can be used to compensate for late-arriving inputs. For example, if a calculation depends on an interface with a known latency of up to 30 seconds, you might adjust this setting to at least 30 seconds to make sure you get the most current event from the interface. Longer settings reduce the chance of missing calculation inputs caused by latency but increase the delay in getting calculation results. Note that calculations use input values at the trigger time stamp when they are evaluated. The default value for this property is 5 seconds.

- **CreateFuturePIPointsForAmbiguousOutputTimes**

  A flag that determines the type of PI point to create for newly mapped output attributes that save output history when the time stamp of the output is uncertain. PI Analysis Service attempts to create the PI point type that matches the time stamp of the output, future PI points for output with future time stamps or historical PI points for output with present or past time stamps. If set to `False`, PI Analysis Service creates historical PI points in cases

when the time stamp of the output is ambiguous. If set to `True`, PI Analysis Service creates future PI points in cases when the time stamp of the output is ambiguous.

- **Default to Auto Create PI Points for Template**

  When this check box is selected, PI Analysis Service automatically creates PI points for output attributes in analyses derived from analysis templates.

- **EvaluationPartitionSize**

  Calculations are grouped for efficient evaluation. Analyses based on an analysis template with periodic scheduling, for example, are evaluated together as one calculation group. To speed up evaluation time, very large calculation groups are partitioned into subgroups based on the **`EvaluationPartitionSize`** setting; the subgroups can then be evaluated in parallel. The default value for this property is 10,000. For a group of analyses with long individual evaluation times, you may want to specify a smaller partition size. A group of 100 analyses, each performing summary calculations for a day's worth of data, may evaluate faster as 4 subgroups with a partition size of 25, depending on the CPU resources available on the machine.

- **EvaluationsToQueueBeforeSkipping**

  If it takes too long to execute a group of calculations, the queue of waiting calculations may become unmanageable. This setting determines when to skip calculations to avoid running out of resources. At the default setting, if more than 50 evaluations are found, PI Analysis Service starts skipping the oldest evaluations to try to return to its normal processing.

- **IsLoadSheddingEnabled**

  For some applications, accuracy and completeness are critical, and no calculations can be skipped regardless of the delay in completing them. To disable skipping calculations, enter `False` for this setting.

- **IsTelemetryAllowed**

  Enter `False` to stop sending usage data from PI Analysis Service for the Customer Experience Improvement Program. (This does not affect the corresponding setting for program participation you can configure in PI System Explorer.)

- **MaxAllowedAutoRecalculationSpanInDays**

  The maximum time span in days for which an analysis will be automatically recalculated on receiving an out of order event. Any recalculation time span exceeding this limit will not be considered for automatic recalculation. The default value for this parameter is 180 days.

- **MaxCacheEventsPerAttribute**

  Once this limit is reached, the oldest events are trimmed from the data cache. Note that this setting takes precedence over the **`CacheTimeSpanInMinutes`** setting.

- **MaxConcurrentRecalculationRequests**

  The maximum number of backfilling or recalculation operations that the PI Analysis Service can perform concurrently. PI Analysis Service dynamically scales the number of threads used for backfilling or recalculation between 1 and the number specified. This number is user configurable and depends on available resources (CPU and memory) of the machine running PI Analysis Service. You can set this parameter to any integer value between 1 and 4*P, where P represents the number of processors on the machine that PI Analysis Service is running. The default value for this parameter is (1/2)*P.

- **MaximumAllowedAutoBackfillingSpanInHours**

  Specify the maximum downtime (in hours) for the PI Analysis Service to initiate automatic backfilling. Setting the parameter to "0" or a negative number will restore the previously recorded setting for this option. If you update this setting, you must restart PI Analysis Service.

  > **Note:**
  > To disable automatic backfilling, you must disable the **`AutoBackfillingEnabled`** parameter.

- **MinCacheEventsPerAttribute**

  The default setting of 1 ensures that at least one event per attribute remains in the data cache.

- **NumberDataWriterThreads**

  The number of threads PI Analysis Service uses to write analysis outputs. More threads are useful for high frequency calculations that write values at high rates and for writing to PI Data Archive on a computer with high network latency (other threads can write values while waiting for a response from PI Data Archive). Consider increasing the number of threads when analyses skip many evaluations or analyses have high evaluation lag. More threads can help if data writes cause a bottleneck. You might set the number of threads to the number of processors on the computer where PI Analysis Service runs.

- **NumberEvaluationThreads**

  The number of threads PI Analysis Service uses for evaluating analyses concurrently. You can set this parameter to any integer value between 1 and 4*P, where P represents the number of processors on the machine that PI Analysis Service is running. The default value for this parameter is P. Consider increasing the number of evaluation threads when PI Analysis Service is not fully utilizing available CPU resources, for example, when analyses skip a number of evaluations or have high evaluation lag. This may happen when you have a lot of analyses with summary calculations that are triggered frequently. Having a larger number of evaluation threads is helpful for analyses that require frequent access to massive amount of input attribute data from outside the data cache.

- **NumberParallelDataPipes**

  Attributes receive updates from data sources, including PI Data Archive, through data pipes. Using a single data pipe to update a large number of attributes may take an unacceptably long time. You can increase the number of data pipes which can operate simultaneously to decrease update time. This can be especially useful for calculations with high-frequency data. Keep in mind, however, that increasing the number of data pipes increases the load on PI Data Archive. If you update this setting, you must restart PI Analysis Service.

- **RuntimeStorageFolderPath**

  Specify the location of the directory that contains PI Analysis Service run-time information such as user backfilling and calculation groups . By default, PI Analysis Service stores run-time information in `\\ProgramData\OSIsoft\PIAnalysisNotifications`.

- **Suggested PI Point Name**

  You can modify the naming pattern here with a substitution parameter. For a complete list, see List of PI AF substitution parameters. Note, however, that some parameters may not be meaningful in the context of a PI point name.

Buffering PI point outputs for PI Analysis Service

You can set up buffering for PI point attributes that write outputs to PI Data Archive from PI Analysis Service. This way, you can optimize throughput and create data resiliency by preventing data loss. You can access Buffering manager from PI System Explorer under **Tools** > **Buffering Manager**. See PI Server topics "Buffering" in Live Library (https://livelibrary.osisoft.com) and "Configure buffering for analysis outputs to PI points" in Live Library (https://livelibrary.osisoft.com) for more information.

Notifications

Notifications is the feature in PI Asset Framework (PI AF) that you use to create and manage notification rules. Notification rules are the mechanism by which users are alerted for response in real time, for conditions that signal events of interest. Notification messages may be customized to include information that is specifically relevant to the event, and may be sent via email to individuals, groups, or to a web service. Further, escalations are also configurable in the notification system.

Within PI System Explorer, you can configure and manage notification rules from the **Notification Rules** tab (visible after you select an element), and from the **Management** plug-in.

> **Note:**
> You can open the audit log directly from notification rules by right-clicking the notification rule and selecting **Audit Trail Events**. For more information on audit trail, see Track PI AF changes with Audit Trail.

Topics in this section

- Introduction to notifications
- Notifications and event frames
- Notifications history
- Requirements for notifications
- Client support for notifications
- Notifications security
- Create a notification rule
- Create a notification rule template
- Trigger criteria in notification rules
- Delivery formats in notification rules
- Subscriptions in notification rules
- Configuration of notifications delivery endpoints
- Contacts
- Configuration of notification rules for analyses or event frames
- Management of notification rules
- Changes from PI Notifications 2012
- Migration from PI Notifications 2012
- Additional resources

Introduction to notifications

Notification rules: Definition and scope

Notifications is included as a feature of PI Server, starting with the 2016 R2 version, and is not installed as a separate product. The notifications feature allows you to configure "notification rules" to send email messages or to call a web service; you can also configure message content and set up escalations.

> **Note:**
> Notifications created using PI Notifications 2012 or earlier are henceforth referred to as "legacy notifications".

Notifications are alerts that are generated when an event of interest is detected in the PI Server; such events are detected by examining and comparing real-time and static data with user-defined conditions configured in the PI Server. User-specified information can be configured in notifications that are sent to users or groups in real time to alert users of conditions to which they may need to respond. Notifications integrate with, and leverage, PI AF and other PI System services, and can be shared, managed, and maintained enterprise-wide. All notification actions, such as notification send times, acknowledgments, entry of comments, and escalations, are stored for later retrieval and examination.

Notification rules are based on event frames and notification messages can include information from event frame templates. For more information, see Notifications and event frames.

PI Notifications Service

PI Notifications Service is a Windows service that evaluates notification rules, processes incoming real-time event frames, and sends notifications. The PI Notifications Service need not run on the same machine as the PI Data Archive, PI AF server, or the client applications.

For information on event frames, see Event frames in PI AF.

Training video

For more information on using notifications in PI Asset Framework, watch the videos on the OSIsoft Learning channel playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2Jbcs43wQ2qO2dxW72PqGhZjKW&rel=0

Notifications and event frames

You can create notification rules based on any generated event frames (regardless of the method used to create them). The notification rules will be triggered when the configured condition or event is detected in the system or process, and a customized notification message is sent to a user, a group, or a web service.

Notification triggers are based on event frames; the event frames may come from different sources like analyses, the Event Frame Generator (EFGen) interface or custom AF SDK applications. Relevant information for events are stored in event frames and compatible tools can retrieve such information. Information contained in elements, event frame templates, and other places may also be added to the notification content. Further, event frame annotations contain historical notification information and may also include attachments; all annotations may be viewed using PI Vision or other client tools that support viewing annotations.

1. Analysis-generated event frames. For more information, see Event frame generation analyses to create analyses from the **Analyses** tab, and Trigger criteria in notification rules to create analyses from the **Notification Rules** tab of PI System Explorer.
2. Event Frame Generator (EFGen) interface-generated event frames. See Event Frame Generator documentation in Live Library (https://livelibrary.osisoft.com).
3. Custom AF SDK applications-generated event frames. See AF SDK documentation on the OSIsoft Tech Support website (https://techsupport.osisoft.com/Documentation/PI-AF-SDK/html/1a02af4c-1bec-4804-a9ef-3c7300f5e2fc.htm).
4. Manually created event frames.

Relationship between event frames and notification rules

Notifications history

Information associated with previous times that a notification was generated is stored as annotations on each event frame that triggered the notification. The annotations can be accessed by selecting the specific event frame of interest from the Event Frames plug-in on PI System Explorer. Each event frame annotation includes such information as the notification rule corresponding to the send event, the subscribers who were notified and when they were notified, and whether the attempt to notify was successful.

If you have migrated from PI Notifications 2012, and elected to migrate both configuration and history, your legacy notifications history is fully migrated; the historical information is migrated as annotations on the event frames created during migration. The migration report includes information on the legacy history instances that were processed during migration.

Note:
For information on permissions required during migration, see Migration from PI Notifications 2012.

Event frame annotations may also be viewed using PI Vision or other client tools that support viewing annotations.

Acknowledge a notification

In PI AF 2016, you can require that an event frame be acknowledged. The acknowledgement feature is used by notifications and PI AF client applications, such as PI Vision. The event-frame acknowledgement feature replaces the acknowledge notification functionality in legacy PI Notifications 2012 and older versions.

For more information on acknowledging event frames in PI AF, see Acknowledgement of event frames and Acknowledge event frames.

For more information on viewing and acknowledging event frames using PI Vision, see the PI Vision topic "View event details and annotate events" in Live Library (https://livelibrary.osisoft.com).

Requirements for notifications

Before you begin, make sure that you have installed PI AF Server 2017 R2.

To use the notifications feature, you must install, in this order:

1. PI AF Client 2017 R2
2. PI Notifications Service 2017 R2
3. Optional. PI Vision.
   - This is required if you want to use the acknowledgement link in the notifications feature, which is a link to the event details page in PI Vision.
   - To include a **Screenshot** from PI Vision in notifications, NTLM provider for Windows Authentication needs to be enabled in PI Vision. For more information, see Delivery formats in notification rules. Also see PI Vision topics "Enable Kerberos delegation using a custom PI Vision service account" in Live Library (https://livelibrary.osisoft.com) and "Enable Basic authentication" in Live Library (https://livelibrary.osisoft.com).
4. Management plug-in from PI AF Client 2017 R2 which allows for managing analyses and notification rules in bulk.

Client support for notifications

PI Vision, PI Coresight 2016 and later versions, and PI DataLink 2016 and later versions support notifications with the ability to display event frames and associated data. Notification messages can include links and screenshots to pre-configured PI Vision or PI Coresight displays, and thereby provide visual representation of the event that triggered the notification. See PI Vision, PI Coresight or PI DataLink product documentation for details on notifications feature support.

Notifications security

The notifications feature uses Windows security for communicating internally and with the PI AF server.

> **Note:**
> See the PI Server topic "PI AF objects and local permissions for PI Notifications Service" in Live Library (https://livelibrary.osisoft.com) for information on PI Notifications Service account permissions.

You can secure access to notification rules, notification rule templates, contacts, and contact templates. Object permissions are granted according to the PI AF Security model.

If Active Directory (AD) access is configured, information from the AD is accessed to create individual contacts. You cannot delete an AD contact from PI System Explorer, and you cannot modify security settings for the contact. To configure AD access, see the PI Server topic "Configure Active Directory objects for delegation" in Live Library (https://livelibrary.osisoft.com).

If you are using a screenshot feature in your email delivery format, see the installation topic "Set up a whitelist of trusted sites to take screenshots from" in Live Library (https://livelibrary.osisoft.com) on security guidelines. For more information, see also Add a screenshot in notification email delivery format and Add a domain to the whitelist.

In line with PI Asset Framework, Notifications supports Windows environments that enforce the use of U.S. Federal Information Processing Standards (FIPS) cryptographic algorithms. For more information, see **Support for FIPS** section in PI System Explorer and Microsoft Support article "System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" security setting effects in Windows XP and in later versions of Windows (https://support.microsoft.com/en-us/help/811833/-system-cryptography-use-fips-compliant-algorithms-for-encryption--has).

Topics in this section

- Security for contacts
- Security for notification rules and templates
- Configure authentication for SMTP server connection
- Configure authentication for a web service connection

Security for contacts

Microsoft Windows security is used to control access to groups, escalation teams, and custom, non-Active Directory contacts. Security settings for Active Directory contacts cannot be changed using PI System Explorer.

Newly created contact objects will inherit default permissions from the parent collections; a newly created custom contact will have security defined by the server's contacts collection.

Topics in this section

- Summary of permissions for contacts
- Edit security for existing contacts

Summary of permissions for contacts

Windows security must be set appropriately for contacts to view, modify, delete, or assign privileges to other contacts.

Permission	Description
`Read`	Can view information on contacts.
`Write`	Can modify contacts.
`Delete`	Can delete contacts.
`Admin`	Can assign privileges to contacts.

Edit security for existing contacts

Edit security settings from the **Contacts** plug-in to allow desired access to contacts. Administrator access is required to edit contact security. See Summary of permissions for contacts.

Note:
You can also edit security configurations for contacts by navigating to **Database** > **Edit Security**, and then selecting the appropriate **Notification Contact Templates** for your PI AF server (from the **Item** column). For example, if your PI AF Server is named MyServer, you must select **MyServer-Notification Contact Templates**.

Procedure

1. In PI System Explorer, click **Contacts** in the navigator pane.
2. Search for the group, escalation team or delivery endpoint whose permissions you want to change. You may need to double-click the contact to see its delivery endpoints.
3. Right-click the contact's delivery endpoint and select **Security**.

   The **Security Configuration** window for the delivery endpoint appears.
4. Change security settings for the delivery endpoint. Permissions can be changed for all identities that have been configured. You can also map new identities and configure their permissions.

Security for notification rules and templates

Notifications uses Microsoft Windows security to control access to notification rules and templates. To edit a notification rule or template, you need write access to notification rules.

Topics in this section

- Summary of permissions for notification rules and templates
- Edit security for a notification rule or template

Summary of permissions for notification rules and templates

Windows security must be set appropriately to perform specific tasks with notifications.

Permission	Description
`Read`	Can view a notification rule.
`Write`	Can modify settings in the Trigger Criteria, Subscriptions, and Formats panes.

Permission	Description
`Subscribe`	Can subscribe oneself to a notification and customize the subscription.
`SubscribeOthers`	Can subscribe others to a notification and customize their subscriptions.
`Delete`	Can delete a notification rule.
`Execute`	Can start, stop or reset a notification rule.
`Admin`	Can assign privileges to notification rules.

Write permission at the PI AF server level is needed for global formats. See Configure security for a PI AF server for more information.

Edit security for a notification rule or template

Edit security settings to allow desired access to notification rules and templates.

Note:

Administrator access is required to edit notification rule and template security. See Summary of permissions for notification rules and templates.

Procedure

1. In PI System Explorer, navigate to **Database** > **Edit Security**.

   You see the **Security Configuration** window.

2. From the **Item** column, select the appropriate notification rule or notification rule template for your PI AF server. For example, on a PI AF server named "MyServer", you must select:
   - **MyServer-Database-Notification Rule** to modify notification rule permissions
   - **MyServer-Database-Notification Rule Template** to modify notification rule template permissions.
3. Change security settings for notification rules or templates. Permissions can be changed for all identities that have been configured. You can also map new identities and configure their permissions.

Configure authentication for SMTP server connection

Best practice is to use a secure connection when using basic authentication. To use a secure connection with an SMTP server, enable the `Use TLS` option. The server must present a valid certificate for a secure connection to be established. To configure authentication for SMTP server connection, follow this procedure.

Procedure

1. To start PI System Explorer, click **Start** > **Programs** > **PI System** > **PI System Explorer**.
2. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
3. Click the **Plug-Ins** tab.

4. Scroll down to the **Delivery Channel Plug-Ins** section, then right-click **Email** and select **Settings**.
5. In the Email Delivery Channel Configuration window, click **Authentication Options** link under **SMTP Server**.

   Note:
   The same can be done for the Backup SMTP Server by clicking the link by the same name under it.

   Primary SMTP Server Authentication Options window opens.
6. Choose among the 3 authentication options.
   - Windows: PI Notifications Service will pass the network credentials of its service account to the SMTP server
   - Anonymous: no credentials will be provided to the SMTP server
   - Basic: enter a username and password the service will provide to the SMTP server

     Note:
     New username and password you put in will override existing credentials saved in the PI Notifications Service.
7. Click **Save and Close** or **Cancel**.
8. Optional. Check **Use TLS** box to enable encryption with basic authentication. Otherwise, the username and password will be sent in plain text.

Configure authentication for a web service connection

Best practice is to use a secure connection when using basic authentication. To use a secure connection for a web service, use the `https://` URI scheme. The server must present a valid certificate for a secure connection to be established. To configure authentication for a web service connection, follow this procedure.

Procedure

1. Go to **Contacts** plug-in.
2. Under **Delivery Endpoints**, select the web service delivery endpoint you want to authenticate.

   Note:
   - Both SOAP and REST delivery methods are supported.
   - Be sure to check in the delivery endpoint if it is newly configured.
3. Click a hyperlink next to **Authentication Option** at the bottom of the window. **Select Authentication Option** window opens.
4. Choose among the 3 authentication options.
   - Windows: PI Notifications Service will pass the network credentials of its service account to the web service
   - Anonymous: no credentials will be provided to the web service
   - Basic: enter a username and password the service will provide to the web service

> **Note:**
> New username and password you put in will override existing credentials saved in the PI Notifications Service.

5. Select **Basic** radio button.
6. Enter the user name and password and click **Save and Close**.

Create a notification rule

Configuring a notification rule includes specifying trigger criteria, adding subscribers to the notification rule, and formatting the message to suit the needs of your organization.

For a detailed sample, see Configuration of notification rules for analyses or event frames.

Procedure

1. In PI System Explorer, select the element on which you want to create notification rules.
2. From either the **Notifications Rules** tab, or from an existing event frame analysis, select **Create a new notification rule**.
3. Enter a name for the new notification rule and (optionally) select a category.
4. In the Trigger Criteria pane, specify the set of conditions that causes a notification to be sent. For more information, see Trigger criteria in notification rules.
5. In the Subscriptions pane, select **Manage Formats** and specify the format for notifications. For more information, see Delivery formats in notification rules.
6. In the Subscriptions pane, select **View/edit subscriptions** and specify the contacts to which notifications will be sent. For more information, see Subscriptions in notification rules.
7. Test that the notification is triggered when an event occurs that satisfies all of the trigger criteria specified.

Create a notification rule template

You can create a new notification rule template for an existing element template in PI System Explorer. You can also convert an existing notification rule into a notification rule template by right-clicking on the notification rule and selecting **Convert to Template**.

For a detailed sample, see Configuration of notification rules for analyses or event frames.

Procedure

1. In the navigator pane of PI System Explorer, click **Library**.
2. Select **Templates** > **Element Templates**.
3. Select the specific element template where you want to create the notification rule template.
4. Click on the **Notification Rule Templates** tab. The tab lists the notification rule templates already defined for the selected element template.
5. Create a new notification rule template for the selected element template. If no notification templates exist, click **Create a new notification rule template** to create the first one. Otherwise, click the New Notification Rule Template icon to create a new notification rule template.

6. Enter information to identify the notification rule template.
   - **Name**

     The name that uniquely identifies this notification rule template.
   - **Description**

     Optional text that describes the notification rule template.
   - **Categories**

     Optional category that you can assign to the notification rule template. Click the list and select the check box next to the category you want to assign, or click **New** to create a new category.
7. Configure trigger criteria for the notification rule template. For specific instructions, see Trigger criteria in notification rules.
8. Click on **Manage Formats** to customize the message that is sent with the notification. For specific instructions, see Delivery formats in notification rules.
9. Click on **View/Edit Subscriptions** to configure subscriptions. For specific instructions, see Subscriptions in notification rules.
10. Click **Check In**

to save the changes.

Trigger criteria in notification rules

You can configure trigger criteria for an event to determine when a notification will be sent to the appropriate delivery channel, for example, an email address or web service. The trigger conditions are configurable in the **Trigger Criteria** pane, and further options can be configured in the Options pane.

To see the Trigger Criteria and Options panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **View/Edit Trigger** in the Trigger pane.

Trigger Criteria pane

The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**.

Use analysis mode to trigger a notification rule on event frames generated by a particular analysis. In the **Analysis** mode, you can select an existing analysis or configure a new analysis of type **Event Frame Generation** or **SQC**, and optionally configure additional trigger conditions based on event frame attribute values. If you choose to create a new analysis with either the default or a custom name, you see the same analysis creation window as from the **Analyses** tab. For information on configuring an SQC analysis, see Create an SQC analysis. For information on creating an Event Frame Generation analysis, see Create an event frame generation analysis. For information on expression functions to be used as a trigger condition, see Expression functions reference.

Note:

- For notifications to take effect, you must check in both the analysis and the notification rule.
- When creating an SQC analysis from the **Trigger Criteria** pane, you can only choose **Event Frame** as the analysis output.

Use event frame search mode to trigger a notification rule based on event frame name, template and category. In the **Event Frame Search** mode, you can select a configured event frame template, from the drop-down list, and then configure the name and category for the event frames that will trigger your notifications. The name may contain wildcard characters that are supported by event frame search. You can optionally configure additional trigger conditions based on event frame attribute values.

For both **Analysis** and **Event Frame Search** modes, you can add additional trigger criteria using event frame attribute values. Criteria can be specified for any event frame template attribute that is specified in the notification rule trigger criteria. The attribute values that you select in the conditions must be specifically those for which you want to be notified of event frames (that match the analysis event frame template). For example, if your event frame template defines an event like "downtime" but you only want an email about "unplanned" downtime, you can configure an attribute value condition where a "reason code" attribute on the "downtime" event frame template has a value indicating "unplanned" downtime.

Note:
Multiple conditions are grouped using the `AND` operator.

Options pane

The Options pane allows you to configure these trigger options:

- **Resend Interval**

  The time interval after which PI Notifications Service will send additional alerts until the event frame matching the notification rule is acknowledged or closed.

- **Non-repetition Interval**

  The time interval during which PI Notifications Service will not send similar alerts associated with the same notification rule.

- **Event Frame Can Be Acknowledged**

  Option to enable event frame to be acknowledged; the event frame template is also modified accordingly. This option is automatically selected if the event frame template has been configured for acknowledgement.

- **Severity option**

  This option applies only to event frame generation analyses. If you have configured multiple start triggers for your analysis, you may choose to be notified in these ways:

  - When the current trigger severity is higher than any trigger severity encountered so far.
  - When the current trigger severity is higher than the previous trigger severity.
  - Every time a trigger condition is true, regardless of its relative severity to other previous triggers.

Topics in this section

- Non-Repetition Interval in notification rules
- Resend Interval in notification rules

Non-Repetition Interval in notification rules

You can configure the **Non-Repetition Interval** setting to prevent notifications from sending similar alerts, which are associated with the same condition, within a specified amount of time.

Alerts will be sent for triggering events with a higher severity even if a prior alert has been sent for a triggering event of lower severity, and within the time-frame of the non-repetition interval.

The system determines when to resend the notification based on the time the last notification was sent, and not based on the start time of the triggering event. For example, if the event frame started at time `t`, and the service, after processing, sends the notification at time `t1`, the system uses `t1` as the basis to calculate the time between alerts.

> **Note:**
> If you specify both a resend interval and a non-repetition interval, the resend interval must be longer.

Resend Interval in notification rules

You can configure the **Resend Interval** setting so that additional alerts are sent when the event frame has not been acknowledged or closed; it is a reminder that the event of interest leading to the notification is yet to be resolved. The resend interval specifies the time between sending successive alerts.

The system determines when to resend the notification based on the time the last notification was sent, and not based on the start time of the triggering event. For example, if the event frame started at time `t`, and the service, after processing, sends the notification at time `t1`, the system uses `t1` as the basis to calculate the time between alerts.

> **Note:**
> If you specify both a resend interval and a non-repetition interval, the resend interval must be longer.

Delivery formats in notification rules

Message and **Content** panes in PI System Explorer

From the Message and Content panes, you can configure the format of the message that is delivered when a notification rule is triggered.

To see the Message and Content panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **Manage Formats** in the **Subscriptions** pane.

Global default formats

The system provides one global format that is used as the default format when a subscriber is added to a notification rule. You can edit or rename the global format, but you cannot delete it.

You can also create additional global formats. These global formats can be viewed in notification rules or notification rule templates. To create, edit or delete global formats, navigate to **Tools** > **Global Formats** in PI System Explorer. For more information on global configuration parameters, see the OSIsoft Tech Support article KB01630 - Manually Managing Global Configuration Parameters in Notifications 2.x (http://techsupport.osisoft.com/Troubleshooting/KB/KB01630/).

Custom delivery formats

To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon. The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names.

To customize your delivery format, you can drag and drop the following types of information from the Content pane to the Message pane:

- PI AF Server properties
- Database properties
- Notification Rule properties
- Event frame properties
- Event frame attributes. If an event frame template is selected, you may add the event frame template attributes to the content of your email notification.
- Referenced element properties
- Referenced element attributes

These properties and attributes can provide a lot of input to analyzing the event of interest; for example, including the event frame start time property can tell you when the event of interest started. Including specific event frame and referenced element attributes can give you supporting information about key parameters.

For PI Vision users, notifications in PI AF 2017 R2 gives the option to construct a link that specifies a full path to a display. You can select **Full Path** from **Referenced element properties** or **Referenced element attributes** under the Content pane.

Note:

- PI Notifications Service and PI System Explorer 2017 R2 or later are required to use the **Full Path** option.
- **Full Path** can also be configured for global delivery formats.
- The existing **Path** option is relative to the reference element or attributes.

When sending a notification email, you can include a web, PI Vision URI, or a file link. That way, you can easily navigate to specific locations for relevant information about attributes, event frames, and so on. To add a link, click in the Message pane, and then click the link icon. You will be prompted to enter your base address. When you click the **Continue** button, **Create a link** window will open. There, you can configure your link.

Notifications in PI AF 2017 R2 has a new link-related feature. You now have an option to include a **Screenshot** (or a static image) of the data that triggered the notification. This way,

separate data retrieval from the source is not necessary, which would have been the case if only a URI (link) was included in the email. You can also specify display options if you choose to display the link as a screenshot. See Add a screenshot in notification email delivery format.

Note:

- Screenshots should come from a trusted source. For more information, see Add a domain to the whitelist.
- To add a **Screenshot** from PI Vision, make sure that NTLM provider is enabled for Windows Authentication in PI Vision. For more information on system requirements, see Requirements for notifications.

You can edit the message if you choose to display your link as a text. **Display link as** option can be found in 2 other places: in **Event Details Hyperlink** (if PI Vision is configured) and URI hyperlink (under **Referenced element attributes**; see Create URI Builder data references for more information).

If you are configuring a Web URI, you can specify a **fragment**, which is a string that is appended at the end of the URI. For more information on query string parameters, see the PI Vision topic "URL parameters reference" in Live Library (https://livelibrary.osisoft.com).

You can add a file attachment to your delivery format by clicking on the **+** symbol in the **Attachments** field. The selected attachment will be uploaded as an attribute on the referenced element. For acceptable types of attachment files, see `/FileExtensions` in AFDiag utility parameters.

Tip:
If the Content pane has a message indicating that your selected format is "read-only", you must first add a new format before you can customize it.

Training video

For more information on message formats, watch the OSIsoft Learning channel video:

https://www.youtube.com/watch?v=2q3YBcaNlEM&rel=0

Add a domain to the whitelist

Protect your system from unwanted security incidents by allowing screenshots only from a trusted source. To add a website to the trusted-sites list, follow this procedure.

Procedure

1. Click **Database** on the PI System Explorer toolbar.
2. Select **Configuration** database.
3. Click **OSIsoft** > **PIANO** in the Elements pane.
4. Go to **Attributes** tab and select **PageRender** > **TrustedSites**.
5. Add the sites in the **Value** field of the attribute delimited by space. Wildcard (*) character is allowed. Here are some examples:

   Note:
   Whitelisting applies to the website and all of its embedded elements (images, scripts).

If you entered ...	The system allows ...	The system blocks ...
*	All websites	No website
https://*	All websites using the https://URI scheme	All other websites
https://*.osisoft.com	https://www.osisoft.com https://techsupport.osisoft.com https://www.osisoft.com/index.html https://www.osisoft.com:80/index.html	http://www.osisoft.com www.osisoft.com https://osisoft.com
www.osisoft.com/index.html	www.osisoft.com/index.html www.osisoft.com/index.html?abc=123	www.osisoft.com/index www.osisoft.com/index.htmlabc
www.osisoft.com/abc	www.osisoft.com/abc www.osisoft.com/abc/def www.osisoft.com/abc.html www.osisoft.com/abc?def=123	http://www.osisoft.com/abcdef www.osisoft.com/abc.def
*.osisoft.com	www.osisoft.com www.osisoft.com:80 www.osisoft.com/index.html techsupport.osisoft.com https://www.osisoft.com http://www.osisoft.com/index.html	osisoft.com notosisoft.com www.osisoft.com.baddomain
osisoft.com	www.osisoft.com www.osisoft.com:80 www.osisoft.com/index.html techsupport.osisoft.com https://www.osisoft.com http://www.osisoft.com/index.html	notosisoft.com osisoft.com.baddomain
https://localserver/PIVision	https://localserver/PIVision https://localserver/PIVision/#/Displays	https://localserver/ https://localserver/PI http://localserver/PIVision https://localserver.int/PIVision

If you entered ...	The system allows ...	The system blocks ...

`*://*.int`	`http://internal.website.int`
	`http://internalwebsite.int`
	`http://internal.website.int:80`
	`http://internal.website.int/index.html`
		`https://www.osisoft.com`
		`http://www.osisoft.com`

`192.168.1.*`	`192.168.1.123`
	`192.168.1.123/abc.html`
	`http://192.168.1.123:8080/abc.html`
		`192.168.22.123`

Add a screenshot in notification email delivery format

Notifications in PI AF 2017 R2 has a new link-related feature. You now have an option to include a **Screenshot** (or a static image) of the data that triggered the notification. To add a screenshot in notification email delivery format, follow this procedure. For more information on security, see Add a domain to the whitelist and installation topic "Set up a whitelist of trusted sites to take screenshots from" in Live Library (https://livelibrary.osisoft.com).

Procedure

1. Navigate to the element that you want to receive the notification.
2. Click on the **Notification Rules** tab.
3. Select the notification rule that you want to add a screenshot. Click **Create a new notification rule** if you are creating a new notification rule.
4. Click **Manage Formats** in the Subscriptions pane.
5. Copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon .
6. Click in the Message pane, and then click the link icon.
7. Enter your base address in Paste a Link window and click **Continue**.
8. In the **Display link as** row, click the **Screenshot** radio button.
9. Optional. In the **Image options** row, choose **Format** and **Size** from each drop down list.
10. Click **OK**.

Subscriptions in notification rules

Contacts and Subscriptions panes in PI System Explorer

From the Subscriptions pane, you can configure the subscribers to be alerted when a notification rule is triggered. A subscriber is an individual entity or group of entities that can receive notification messages by subscribing to notification rules.

To see the Contacts and Subscriptions panes on PI System Explorer, navigate to the **Notification Rules** tab for a selected element, and then click **View/Edit Subscriptions** in the Subscriptions pane.

To add a subscriber from the **Notifications Rules** tab of PI System Explorer, simply drag and drop a contact from the Contacts pane to the Subscriptions pane.

> **Note:**
> To unsubscribe a contact, right-click on the contact and select **Unsubscribe**.

When you install PI AF, information from Active Directory (AD) is imported and individual contacts are automatically created in Contacts. For more information, see Configure Active Directory access for contacts. If a user does not have an AD account, you can create a custom contact.

Subscriptions pane

When a contact is added to the Subscriptions pane, the following details are included for the contact:

- **Name**

  The name of the contact.

- **Configuration**

  The default, inherited, or custom delivery format for this subscriber.

- **Notify Option**

  The notify option may be one of:

  - Event start and end. You can receive a notification when an event frame matching the notification rule is created or closed.
  - Event start.
  - Event end. You can receive a notification only when an event frame is closed.

Contacts pane

Subscribers may be configured from these types of contacts on the Contacts pane:

- Individual contact
- Escalation teams
- Groups
- Delivery endpoints

Use **Contacts Search** to find individual contacts to include in a group or escalation team, or subscribe to a notification rule. You may use wildcards to expand your name searches; this will search both first and last names. For example, searching for "A*" will list both "Adam Smith" and "John Adams".

> **Note:**
> The search results show the display name as configured in Active Directory.

For more information on configuring contacts using the Contacts plug-in, see Contacts.

Training video

For information on how to configure subscribers, watch this video:

https://www.youtube.com/watch?v=lN2r5u5_U2w&rel=0

Customization of subscription content and delivery

The default delivery format specified on the **Message** tab is automatically applied to all subscriptions in the notification. To send a different message, you can customize subscription formats by applying a different delivery format.

> **Note:**
>
> If you customize subscriptions and later want to change delivery formats, you might need to manually change the delivery formats of subscriptions that you previously customized. For example, if you change the default delivery format, or specify a different delivery format as the default, only subscriptions that inherit that format will be changed; customized subscriptions are not affected.

For more information on delivery formats, see Delivery formats in notification rules.

Configuration of notifications delivery endpoints

A delivery endpoint is a single entity to which a notification is delivered, such as an individual email address or web service call. Each delivery endpoint is associated with a delivery channel, the conduit through which notification messages are sent. Notifications provides delivery channels for email and for both SOAP and REST web services.

To configure email delivery endpoints, see Configure an email delivery endpoint. To configure dynamic email delivery endpoints, new in PI AF 2017 R2, see Configure a dynamic email delivery endpoint.

To configure web service delivery endpoints, see Configure a REST web service delivery endpoint.

To configure notifications to be sent as text messages, see the OSIsoft Tech Support article KB00294 - How do I set up PI Notifications to send text messages to my cellular phone (https://techsupport.osisoft.com/Troubleshooting/KB/KB00294).

Topics in this section

- Configure an email delivery endpoint
- Configure a dynamic email delivery endpoint
- Configure a REST web service delivery endpoint

- Configure a SOAP web service delivery endpoint

Configure an email delivery endpoint

To configure an email delivery endpoint and add an email subscriber to your notification rule, follow this procedure.

Before you start

If you have not already configured the SMTP server settings for the email delivery channel, follow the instructions in: Update SMTP settings for the email delivery channel.

Procedure

1. Add an email delivery endpoint using the PI System Explorer Contacts plug-in. For specific information, see Manage contacts and Options for the notifications email delivery channel. See Configure authentication for SMTP server connection for setting up authentication.
2. Navigate to **Elements** > ***your_element*** > **Notification Rules** and add the configured email delivery endpoint to your notification rule as a subscriber.
   a. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule
   b. Drag and drop the configured contact from the Contacts to the Subscriptions pane.

Update SMTP settings for the email delivery channel

You need to configure the email delivery channel to be able to receive email notifications. This must be done once for each PI AF server.

Procedure

1. To start PI System Explorer, click **Start** > **Programs** > **PI System** > **PI System Explorer**.
2. Click on the PI System Explorer toolbar, or click **File** > **Server Properties**.
3. Click the **Plug-Ins** tab.
4. Scroll down to the **Delivery Channel Plug-Ins** section, then right-click **Email** and select **Settings**.
5. In the Email Delivery Channel Configuration window, change the following parameters as needed:
   - **SMTP Server**: The fully-qualified domain name of the machine running the SMTP server.
   - **Port**: The listener port of the SMTP Server.
   - **Authentication options**: Options for a primary SMTP server authentication. See Configure authentication for SMTP server connection.
   - **Use TLS**: Option to enable Transport Layer Security (TLS) encryption for the primary server
   - **Backup SMTP Server**: The fully-qualified domain name of the machine running the backup SMTP Server.
   - **Port**: The listener port of the backup SMTP Server.
   - **Authentication options**: Options for a secondary SMTP server authentication.

- **Use TLS**: Option to enable Transport Layer Security (TLS) encryption for the secondary server
- **Sender Email**: The email address from which notification messages will be sent.
- **Allow contacts to set sender email**: Specifies whether individual contacts can change the email address from which their notification emails are sent.
- **Send Timeout**: Time allowed for sent emails to be received by the primary SMTP server before failover to the backup SMTP server occurs.
- **Backup Fail Back Time**: During failover, specifies how long the backup SMTP server sends email before attempting to fail back to the primary server.

Configure a dynamic email delivery endpoint

In PI AF 2017 R2, you can configure an email delivery endpoint as a value of an attribute. This provides an additional flexibility in situations where notification emails can be sent to different recipients without making changes to the notification rule template. To configure a dynamic email delivery endpoint and add an email subscriber to your notification rule, follow this procedure. For a conventional email delivery endpoint configuration, see Configure an email delivery endpoint.

Procedure

1. Navigate to the element that you want to receive the notifications from and add an attribute with your email address as a value. The value of this attribute must be a string. You can enter multiple email addresses delimited by comma.
2. Navigate to **Notification Rules** tab to configure and add an email delivery endpoint as a subscriber to your notification rule.
   a. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.
   b. Click **Create a new dynamic endpoint** under **Dynamic Endpoints**.
   c. Select an email attribute configured in step 1.
   d. Click **Create**.
   e. Drag and drop the configured contact from the Contacts to the Subscriptions pane.
   f. Click **OK**.
3. Go to **Contacts** plug-in to edit or delete a configured delivery endpoint. It will be found in **Delivery Endpoints** folder.

   Note:
   - The email attribute endpoint can be edited from the Subscriptions pane as well.
   - **Test** button in the **Contacts** plug-in is not supported for an attribute-based endpoint at this time.

Configure a REST web service delivery endpoint

To configure a web service delivery endpoint and add a REST web service subscriber to your notification rule, follow this procedure. For SOAP web service, see Configure a SOAP web service delivery endpoint.

Procedure

1. Add a web service delivery endpoint using the PI System Explorer Contacts plug-in.
   a. Go to the **Contacts** plug-in of the PI System Explorer.
   b. Right-click the **Delivery Enpoints** folder.
   c. Select **New Delivery Enpoint**.
   d. Optional. **Retry interval**. Select the time interval at which PI Notifications Service will contact the web service. The web service will be contacted as frequently as the system allows if zero is entered.
   e. Optional. **Maximum Retries**. Select the number of times PI Notifications Service will try to connect to the web service. Entering zero means there will be no retries.
   f. Select **WebService** from the drop-down list next to **Delivery channel**.
   g. Select **REST**.
   h. Enter the URL of your web service.
   i. Select **Http Method**, a default web method to be used for the notification. Supported methods are POST, PUT and PATCH.
   j. Select **Authentication Option**.
      - Windows: the default option. PI Notifications Service will pass the network credentials of its service account to the web service
      - Anonymous: no credentials will be provided to the web service
      - Basic: enter a username and password the service will provide to the web service
   k. Check in the changes.

   See **Create a delivery endpoint such as a stand-alone email or a web service** row in Manage contacts and Options for the notifications web service delivery channel for more information. See also Configure authentication for a web service connection for authentication of web service.
2. Add the configured web service delivery endpoint to your notification rule as a subscriber.
   a. Go to the **Notification Rules** tab of the element you want to add the web service as the delivery endpoint.
   b. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.
   c. From the Contacts pane on the right-hand side, expand **Delivery Endpoints** and drag and drop the configured web service to the Subscriptions pane on the left.
   d. Configure the REST web service.

      REST can send a JSON object to the REST server in HTTP methods PUT, POST or PATCH. Click the "wrench" icon to configure the JSON object for the REST API method.

      Currently, only JSON payloads are supported; query string parameters are not supported.

      If you hover over the "Help" icon , you see sample JSON parameters (path and value) that are necessary for configuring this method:

Enter the path and value on each row. You can validate the method using the **Test Send** button.

Configure a SOAP web service delivery endpoint

To configure a web service delivery endpoint and add a SOAP web service subscriber to your notification rule, follow this procedure.

Procedure

1. Add a web service delivery endpoint using the PI System Explorer Contacts plug-in.
   a. Go to the **Contacts** plug-in of the PI System Explorer.
   b. Right-click the **Delivery Enpoints** folder.
   c. Select **New Delivery Enpoint**.
   d. Optional. **Retry interval**. Select the time interval at which PI Notifications Service will contact the web service. The web service will be contacted as frequently as the system allows if zero is entered.
   e. Optional. **Maximum Retries**. Select the number of times PI Notifications Service will try to connect to the web service. Entering zero means there will be no retries.
   f. Select **WebService** as the **Delivery channel**.
   g. Select **SOAP**.
   h. Enter the web service address, the URL of your web service. You can validate the connection using the **Get Web Services** button.
   i. Enter the name of the web service to be used for notification in the **Web Service** field.
   j. **Default Web Method**: select the default web method to be used for the notification. This menu displays all of the parameters defined in the web service.

k. Select **Authentication Option**.

- Windows: the default option. PI Notifications Service will pass the network credentials of its service account to the web service
- Anonymous: no credentials will be provided to the web service
- Basic: enter a username and password the service will provide to the web service

l. Check in the changes.

See **Create a delivery endpoint such as a stand-alone email or a web service** row in Manage contacts and Options for the notifications web service delivery channel for more information. See also Configure authentication for a web service connection for authentication of web service.

2. Add the configured web service delivery endpoint to your notification rule as a subscriber.

a. Go to the **Notification Rules** tab of the element you want to add the web service as the delivery endpoint.

b. Click **View/Edit subscriptions** in the Subscriptions pane of the notification rule.

c. From the Contacts pane on the right-hand side, expand **Delivery Endpoints** and drag and drop the configured web service to the Subscriptions pane on the left.

d. Configure the SOAP web service.

SOAP can call the web service methods. Click the "wrench" icon to configure the SOAP API method. The "Information" icon gives you information about the parameters (path and value) that must be configured for this method, and their complexity. Enter the path and value on each row (the path depends on how the web service server is configured); clicking on a row brings up a drop-down list of expected values for that path. The interface also automatically validates the parameters that you enter on each row. You can validate the method using the **Test Send** button.

You have the option to cancel or retry a connection that is taking a long time to complete; this may happen if you have a high latency (commonly called "laggy") connection where the WSDL information is not quickly retrieved.

> **Note:**
> Complex data may be required by the method. To enter the correct information, refer to the Web Service Definition Language (WSDL) documentation for the API method.

Contacts

In PI System Explorer, the Contacts plug-in provides you with contact options for individual users and groups.

Contacts plug-in versions

PI System Explorer works with different versions of the Contacts plug-in.

- To work with the version that is compatible with the new version of notifications that was released in 2016 R2, execute the 64-bit version of PI System Explorer.
- To work with the legacy version that is compatible with legacy notifications, execute the 32-bit version of PI System Explorer. Note that if legacy notifications is not installed, the new version of the Contacts plug-in will load. For more information on PI System Explorer versions, see PI System Explorer.

Subscriber configuration

You can configure subscribers from these types of contacts:

- **Delivery endpoints**

  A delivery endpoint is an entity to which notifications can be sent, and is typically an individual user or an application.

- **Groups**

  A group contact is an unordered collection of delivery endpoints, other groups or escalation teams. Notification messages are sent to all members of the group simultaneously.

  > **Note:**
  > Group contacts that you create in PI System Explorer are not the same as Active Directory (AD) groups.

- **Escalation teams**

  An escalation team is an ordered collection of delivery endpoints or groups.

  Notification messages are sent immediately to the first contact on the list. If the event frame matching the notification rule is not acknowledged or closed within a specified time, notification messages are sent sequentially to the remaining members of the escalation team until the event frame is acknowledged or closed.

  The escalation period defines the amount of time to elapse before a notification is sent to the next contact on the list.

Search for contacts

Click the icon to open the Search Contacts window in the Contacts pane on the top left of the screen. Currently available search criteria are: Name, Description, Department and Email.

> **Note:**
> You must be running the 2018 or later version of PI AF Server to use email as search criterion. Note that search by email address only works for contacts in Active Directory but does not find contacts created manually.

Topics in this section

- Configure Active Directory access for contacts
- Manage contacts
- Options for the notifications email delivery channel
- Options for the notifications web service delivery channel

Configure Active Directory access for contacts

When you use notifications with PI AF server, you may need to specify how to access Microsoft's Active Directory to retrieve contact names for the PI Notifications Service Contacts lists.

Each PI AF server provides the option to specify the domain and contact sub-folder, as well as the account needed to access Active Directory and retrieve contact names. By default, the account under which the PI AF server application service is running is used for Active Directory access. To use a different account or to access an Active Directory in a different domain, configure access from the Configure Active Directory Access for Contacts window.

Note:

- Beginning with PI AF 2017 R2, Active Directory group is shown by default under **Contacts** in PI System Explorer once it is configured.
- Notifications 2016 R2 or later will automatically handle a change in email address of the user or a group in Active Directory by contacting the domain controller before sending an email each time.

Procedure

1. Open PI System Explorer and connect to a database on the PI AF server for which you want to configure Active Directory access.
2. Click **File** > **Server Properties**.
3. In the PI AF Server Properties window, click the **Configure Active Directory Access for Contacts** link.
4. In the **Active Directory Domain Name** text box, enter the full DNS name of the Active Directory domain from which the contact names will be retrieved for the PI Notifications Service Contacts (for example, `contoso.com`).

   If this field is left blank, the domain in which the PI AF application service resides will be used.
5. In the **Active Directory Contact Sub-Folder** text box, enter the path to the folder containing the list of contacts for this domain.

   In larger Active Directory domains, contacts may be organized within sub-folders. The use of sub-folders can allow for faster retrieval of a list of Active Directory contacts.

   Use the following structure for the sub-folder:
   ```
   DomainUserFolder/SubDomainUserFolder/Sub-SubDomainUserFolder
   ```
6. Choose an option for **Active Directory Access Account**:
   - **Use the account the PI AF Server runs as**

     This is the default option. Select it to access Active Directory using the account under which the PI AF application service runs. By default, the PI AF server is installed using a virtual account, `NT SERVICE\AFService`. However, the PI AF server service account can be changed. If the PI AF server service account does not have the necessary permission to read the Active Directory, no contact names will be retrieved in the Contacts list. If your Active Directory security is configured to allow the PI AF server service account to read the Active Directory, this is the simplest option.
   - **Use the account the AF Client is running as**

Select this option to use the credentials of the user account under which the connecting client application is running. If the PI AF server service is running under an account (a virtual account, `NT SERVICE\AFService` is the default account) that does not have permission to read the Active Directory, this option can be used. As long as the user account under which the connecting client application is running has permission to read Active Directory, a list of contact names is returned to the Contacts list. The contents of the Contacts list may vary, depending upon the access account used, since the security to read the contact list is determined by Active Directory.

> **Note:**
> Specifying this option may require Kerberos configuration if a PI AF SDK application will be using impersonation in a middle tier, such as a Web Service.

- **Use the specified account**

  This option allows you to specify an account to use to read the Active Directory. This can be useful when the Active Directory and PI AF server are in different domains or when the accounts in the first two options have no permission to read the Active Directory. For **Account Name**, use the format `Domain\User`. Make sure the specified account has the appropriate permission to read the target Active Directory.

7. Check **Use Active Directory's locally cached Global Catalog** to use the global catalog for Active Directory domain controller searches. Otherwise searches must go to the owning domain controller.

   Active Directory holds information in a distributed data repository called a global catalog. For installations where there are multiple, distributed domain controllers, each domain controller has a cache of the portions of the global catalog for which it is not responsible, so that Active Directory searches do not have to be referred to the owning domain controller. This improves performance for queries that must otherwise have to access a remote domain controller.

8. Choose a setting for **Return All Persons**.

   Active Directory objects are derived from one another as follows:

   ```
   Top>Persons>OrganizationalPerson>Contact
   ```

   and

   ```
   Top>Persons>OrganizationalPerson>User
   ```

   - Select this check box to return Persons, Organizational Persons, Contacts and Users from the target Active Directory.
   - Clear the check box to return only Users.

Manage contacts

Use the Contacts plug-in to manage individual contact information, as well as manage groups, escalation teams, and delivery endpoints for use with notifications.

Before you start

You cannot nest escalation teams.

Procedure

1. In the navigator, click **Contacts**.
2. In the Contacts browser, choose from the following actions.

To ...	Do this ...
Edit an existing contact	a. Choose one of the following actions: ▪ Enter search criteria in **Search for contacts** and press Enter. Note that * returns all contacts. ▪ Click next to **Search for contacts**, enter search criteria in the Search Contacts window, and click **OK**. ▪ Expand the **Contacts** folder, click **New search...**, enter search criteria in the Search Contacts window, and click **OK**. Search results are displayed in the Contacts browser. b. Click beside the contact you want to update and click the email delivery endpoint. c. In the viewer, you can make the following changes: ▪ In the **Description** field, add additional contact information. ▪ Enter notification contact options in the **Retry interval** and **Maximum Retries** fields.
Create a custom contact that is not already in Active Directory	a. Right-click the **Contacts** folder and click **New Contact**. b. In the viewer, enter a name and an email address. All other information is optional. The contact is displayed in the Contacts browser. c. Click beside the newly created contact. **Note:** The icon indicates non-AD contacts (as opposed to). d. Click the email delivery endpoint and, in the viewer, enter notification contact options in the **Retry interval** and **Maximum Retries** fields.

Create a group	a. Right-click the **Groups** folder and click **New Group**. b. In the **Name** field, enter a unique name for the group. c. Choose from the following actions: ▪ Click and, in the Select a Contact window, locate a contact with a new search, or select another group, an escalation team, or a delivery endpoint, and click **OK**. Repeat as needed. ▪ In the Contacts palette, locate a contact with a new search, or select another group, an escalation team, or a delivery endpoint, and drag it onto the viewer. Repeat as needed. d. Set notification contact options for the group. ▪ Right-click each subscriber delivery endpoint, click **Options**, and enter notification contact options in the **Retry interval** and **Maximum Retries** fields. ▪ If escalation teams are included in a group, right-click each escalation team, click **Options**, and enter notification contact options in the **Escalation period** and select an **If not acknowledged** option.

Create an escalation team	a. Right-click the **Escalation Teams** folder and click **New Escalation Team**. b. In the **Name** field, enter a unique name for the escalation team. c. In **Escalation period**, specify how long you want an escalation to last. d. In **If not acknowledged**, specify the action to be taken if the notification is not acknowledged after being sent to all contacts in a team: ▪ To stop the escalation process, select **End escalation**. ▪ To repeat the escalation process for a specified number of times until the notification closes or is acknowledged, select **Repeat *N* times**. ▪ To repeat the escalation process indefinitely until the notification closes or is acknowledged, select **Repeat while active**. e. Choose from the following actions: ▪ Click and, in the Select a Contact window, locate a contact with a new search, or select another group, or a delivery endpoint, and click **OK**. ▪ In the Contacts palette, locate a contact with a new search, or select another group, or a delivery endpoint, and drag it onto the viewer. f. Configure the sequence for the escalation chain. ▪ Click a subscriber and click and to position as needed. ▪ To remove a subscriber, click the subscriber to be removed and click .

Create a delivery endpoint such as a stand-alone email or a web service	a. Right-click the **Delivery Endpoints** folder and click **New Delivery Endpoint**. b. In the **Name** field, enter a unique name for the delivery endpoint. c. Enter notification contact options for the delivery endpoint in the **Retry interval** and **Maximum Retries** fields. d. From **Delivery channel**, select a deliver option. ▪ For **Email**, enter address configuration information, as described in Options for the notifications email delivery channel. ▪ For **WebService**, enter web service address configuration information, as described in Options for the notifications web service delivery channel.

3. To save changes you have made to contacts, press Ctrl+S or click **Check In**.

Options for the notifications email delivery channel

An email delivery endpoint can be configured in ways specific to the email delivery channel, such as the address to which notification is sent. The following table shows options that are available for all email delivery endpoints:

Option	Description
`To Email`	Email address to which the notification is sent. This option cannot be changed in the default email delivery endpoint that was derived from the Active Directory (AD) contact. Multiple email addresses can be entered delimited by comma.
`From Email`	Email address from which the notification is sent. This option is available solely if your PI System manager has configured the email delivery channel with the setting `Allow contacts to set sender email`.
`Use HTML formatting`	Specifies if the delivery endpoint receives messages in HTML format. HTML formatting makes messages easier to read and allows messages to contain hyperlinks.

Options for the notifications web service delivery channel

Creating a web service delivery endpoint

Using the Contacts plug-in of the PI System Explorer (PSE), you can configure a web service delivery endpoint for SOAP or REST web services. To map parameters for the API call, create and configure body of the message, see Configure a SOAP or REST web service delivery

endpoint. For authentication of a web service, see Configure authentication for a web service connection.

Configuring the SOAP web service

The following table lists the options you must set for a SOAP web service delivery endpoint:

SOAP web service options

Option	Description
`Web Service Address`	The URL of your web service. You can validate the connection using the **Get Web Services** button.
`Web Service`	The name of the web service to be used for notification.
`Default Web Method`	Default web method to be used for the notification. This menu displays all of the parameters defined in the web service.

The PI System Explorer user interface automatically retrieves the associated Web Service Definition Language (WSDL) methods, provides useful information about the parameters, and guides you through configuring the web service.

Configuring the REST web service

The following table lists the options you must set for a REST web service delivery endpoint:

REST web service options

Option	Description
`URL`	The URL of your web service.
`HTTP Method`	Default web method to be used for the notification. Supported methods are POST, PUT AND PATCH.

Configuration of notification rules for analyses or event frames

The asset analytics and notifications features of the PI AF server together provide effective ways for you to perform condition-based or predictive maintenance. Notification rules leverage the event frame feature of PI AF to notify you of important events that affect your system.

This section guides you in creating notification rules for your analyses or event frames.

Topics in this section

- Configure notification rules from analyses
- Configure notification rules for user-defined event frames
- Configure escalations for notification rules

Configure notification rules from analyses

If you have already configured analyses to detect anomalies in your PI AF asset data, or you wish to configure such analyses, follow this procedure to configure notification rules on configured analyses.

Before you start

For information on software and system requirements, see Requirements for notifications.

> **Note:**
> If you want to configure notification rules to trigger on event frames from sources like event frame interfaces or custom applications, see Configure notification rules for user-defined event frames.

For information on creating an event frame generation analysis, see Event frame generation analyses.

Procedure

1. Select the existing event frame generation analysis for which you want to create a notification rule.

2. Click **Create a new notification rule for the selected analysis**. You see the **Notifications Rules** tab highlighted, as well as the options to configure a notification rule with the default name "Notification Rule".

3. Edit the created notification rule.

   Provide one or more of the specified criteria; some of your criteria may be pre-selected from your analysis. The notification rule is triggered when all criteria are satisfied. You may

change the name of the notification rule from the default, add a suitable description, and select a suitable category.

Selecting a category aids in organizing and searching for your notification rules.

4. Click **View/Edit Trigger**

   The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**. Make sure that the **Analysis** criteria mode is selected.

5. Configure options in the Trigger Criteria pane.

   - You see the details of the selected analysis such as the event frame template name, the start and end triggers, and so on.
   - You can configure a new analysis of type **Event Frame Generation** or **SQC**. When you choose to create a new analysis with either the default or a custom name, you see the same analysis creation window as from the **Analyses** tab of PI System Explorer.
   - For more information, see Trigger criteria in notification rules.

6. Configure options in the Options pane.

   - Optional. **Resend Interval**. Select the time interval at which PI Notifications Service will send additional alerts if the event frame matching the notification rule is not acknowledged or closed. For more information, see Resend Interval in notification rules.
   - Optional. **Non-repetition Interval**. Select the time interval for which PI Notifications Service will not send similar alerts associated with the same notification rule. For more information, see Non-Repetition Interval in notification rules.
   - **Event Frame Can Be Acknowledged**. This option is automatically selected if the event frame template has been configured for acknowledgement.
   - Optional. Options based on multiple start triggers. If you have configured multiple start triggers for your analysis, choose between the options of being notified (a) when the current trigger severity is higher than any trigger severity encountered so far, (b) when the current trigger severity is higher than the previous trigger severity, or (c) every time a trigger condition is true, regardless of its relative severity to other previous triggers.

7. Optional. To configure the email format, click **Manage Formats** in the Subscriptions pane. You see the Message and Content panes. By default, the notifications feature uses the Global

Default Email format for each subscriber. For more information, see Delivery formats in notification rules.

> **Note:**
> If the Content pane has a message indicating that your selected format is "read-only", you must first add a new format before you can customize it.

8. Optional. To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon.

   The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names. To add more information to your custom format, drag and drop information from the Content pane to the Message pane.

   > **Tip:**
   > A hyperlink indicates that an event frame template is selected; you may add the attributes defined below the event frame template to the content of your email notification by dragging and dropping the attributes from the Content pane to the Message pane.

   You can designate your custom format as the default by clicking the **Is Default** box in the Message pane.

9. Click **View/edit subscriptions** in the Subscriptions pane.

   You can search for contacts using **Contacts Search** in the Contacts pane. To add a subscriber, drag and drop a specific contact from the Contacts pane to the Subscriptions pane. Make sure to select your customized email format if you have created one, or pick the Global Default Email format. For more information, see Subscriptions in notification rules.

10. Optional. Configure escalation teams.

    You can configure members of escalation teams from the Contacts plug-in on PI System Explorer, as described in Manage contacts. The escalation team contacts or groups will be notified one by one at the end of time segments specified in **Escalation period**. You can also control the frequency of the escalation notification.

    > **Note:**
    > Escalations are triggered on event frame templates that can be acknowledged. Open the **Library** plug-in on PI System Explorer, select your event frame template and verify that the **Event Frame Can Be Acknowledged** check box is selected.

11. Check in the notification rule.

    When the event of interest occurs, an email will be sent to the subscriber's mailbox. The email shows the event frame that triggered the notification rule with customized details that you have configured.

After you finish

To manage your notification rules, use the **Management** plug-in on PI System Explorer. See Management of notification rules.

Configure notification rules for user-defined event frames

Use this procedure if you want to configure notification rules to trigger on event frames from sources like event frame interfaces or custom applications.

Before you start

For information on software and system requirements, see Requirements for notifications.

> **Note:**
> If you have already configured an analysis on your PI AF asset, or wish to create new analyses to generate the event frames, use the procedure Configure notification rules from analyses.

Procedure

1. In the PI System Explorer navigator, click **Elements** and select the element for which you want to define the notification rule.
2. Select the **Notification Rules** tab.
3. Click **Create a new notification rule**.

Change the name of the notification rule from the default, add a suitable description, and select a suitable category. Selecting a category aids in organizing and searching for your notification rules.

4. In the Trigger pane, click on the hyperlink **Please configure trigger criteria for this Notification Rule**.

   The Trigger Criteria pane allows you to configure either of these trigger criteria modes: **Analysis** or **Event Frame Search**.

5. Select the **Event Frame Search** trigger criteria mode.

6. Configure options in the Trigger Criteria pane.

   You can select the event frame template from the drop-down list, and add conditions using any of the attribute templates for the event frame template.

7. Configure options in the Options pane.

   - Optional. **Resend Interval**. Select the time interval at which PI Notifications Service will send additional alerts if the event frame matching the notification rule is not acknowledged or closed.
   - Optional. **Non-repetition Interval**. Select the time interval for which PI Notifications Service will not send similar alerts associated with the same notification rule.

   Note:
   Verify that the event frame template has the **Event Frame Can Be Acknowledged** check box selected.

8. Optional. To configure the email format, click **Manage format** in the Subscriptions pane.

   You see the Message and Content panes. By default, notifications uses the Global Default email format for each subscriber.

   Note:
   If the Content pane has a message indicating that your selected format for a subscriber is "read-only", you must first add a new format before you can customize it.

9. Optional. To create a custom delivery email format, copy the default **Global Default Email** format by clicking on the copy icon in the Message pane. Alternatively, you can create a new delivery format by clicking the new delivery format icon.

   The default format includes basic notification information such as the notification rule name, trigger time and event frame start time, as well as system information such as system and database names. To add more information to your custom format, drag and drop information from the Content pane to the Message pane.

   > **Tip:**
   > A hyperlink indicates that an event frame template is selected; you may add the attributes defined below the event frame template to the content of your email notification by dragging and dropping the attributes from the Content pane to the Message pane.

   You can designate your custom format as the default by clicking the **Is Default** box in the Message pane.

10. Click **View/edit subscription** in the Subscriptions pane.

    You can search for contacts using **Contacts Search** in the Contacts pane. To add a subscriber, drag and drop a specific contact from the Contacts pane to the Subscriptions pane. Make sure to select your customized email format if you have created one, or pick the Global Default email format.

11. Optional. Configure escalation teams.

    You can configure members of escalation teams from the Contacts plug-in on PI System Explorer, as described in Manage contacts. The escalation team contacts or groups will be notified one by one at the end of time segments specified in **Escalation period**. You can also control the frequency of the escalation notification.

12. Check in the notification rule.

    When the event of interest is generated, an email is sent to the subscriber's mailbox. The email shows the event frame that triggered the notification rule with the customized details that you have configured.

After you finish

To manage your notification rules, use the **Management** plug-in on PI System Explorer. See Management of notification rules.

Configure escalations for notification rules

An escalation team is an ordered collection of delivery endpoints or groups. Notification messages are sent sequentially to the members of escalation teams until the event frame matching the notification rule is acknowledged or closed. You can set up an escalation structure to ensure that the right people are notified with the right information, and in the right order.

Subscribe an escalation team to send notification messages sequentially to a group of contacts. For more information on creating an escalation team, see **Create an escalation team** in Manage contacts.

Management of notification rules

To manage your notification rules, use the Management plug-in, accessible on the bottom left of the main screen of the PI System Explorer. When accessed, the plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Notification Rules** button to view and manage your notification rules. For more information, go to Management of notification rules. To search for notification rules configured in your database, see Search for analyses or notification rules.

Changes from PI Notifications 2012

The notifications feature is based on event frames and does not have an analysis engine like PI Notifications 2012 and earlier versions (henceforth referred to as "legacy" versions). Notification rules leverage PI Analysis Service, as well as other applications that generate event frames, to identify important events in the PI Server, to use event frames as the container for relevant information for the event, and to use annotations on the event frame to store notification history.

The notifications feature is driven by event frames; therefore, some legacy notification features are either different or rendered obsolete. A migration tool is included to enable migration of notifications from your PI Notifications 2012 legacy version to notification rules, thereby preserving the investment you made in your legacy system.

> **Note:**
> If you choose to migrate, the migration tool converts your legacy notifications into analyses and notification rules.

For more details on the migration tool, see Migration from PI Notifications 2012.

User interface changes from PI Notifications 2012

Notifications in PI Server 2016 R2 (and later versions) is fundamentally different from the legacy PI Notifications 2012, and you cannot share configurations between the two systems.

To configure event-based notifications, you must use the **Notification Rules** tab in the **Elements** view of 32-bit or 64-bit PI System Explorer. To manage legacy notifications, you must use the **Notifications** plug-in on the 32-bit PI System Explorer.

Migration from PI Notifications 2012

PI Server 2016 R2 and later versions include a migration tool to enable migration of your 2012 notifications to notification rules based on event frames. The migration tool is run at the end of the PI Notifications Service install, and you may choose to migrate your legacy notifications at that time.

You may also run the migration again from PI System Explorer, after the installation is complete. To run the migration tool from PI System Explorer, make sure that all these conditions are true:

- PI Notifications Service is installed on your machine
- The migration tool executable file `PINotificationsMigrationTool.exe` is installed on your machine, typically in the `PIPC\Notifications` folder

The migration tool causes these object migrations:

- Legacy notification triggers are migrated as event frame generation analyses and are run in PI Analysis Service.
- Legacy notifications created in previous versions are migrated to notification rules, and they run in the PI Notifications Service.

- Legacy notifications history is fully migrated and created as event frames. History in PI points are migrated as annotations on event frames.
- Legacy notification template name, description and properties are mapped to the notification and target properties; a message is also logged in the migration report.

> **Note:**
>
> Properties of the notification rule template cannot be overridden on a notification rule based on the template with an exception of the description field.
>
> Due to this, if a legacy notification deviates from the legacy notification template, that information may be lost during migration. For example, if the legacy notification template is named "Notification for Pumps", but a legacy notification based on this template is named "Notification for Pumps - Houston", the migrated notification rule template and all notification rules based on this template will be named "Notification for Pumps". The description field of the migrated notification rule, however, will contain the name of the legacy notification. The migration report will contain the name changes and any other changes that have been made.

Training video

For more information on migration, watch the OSIsoft Learning channel video:

https://www.youtube.com/watch?v=bGdN6VrBVUw&rel=0

Running the migration tool

Before you begin migration you must verify that you have `read` permissions on the PI Data Archive where the legacy notifications history is stored and `read/write` permissions on the PI AF database where the legacy notifications are configured.

You may need to back up the PIFD (SQL database) if you want to re-run the migration and avoid having duplicate notification rule and analysis objects created. If you re-run your migration and are warned that duplicates may be created during the re-migration, you can first restore your PIFD database from the backup and avoid the creation of duplicates. For instructions on how to perform a backup of the PIFD database, see the PI Server topic "OSIsoft Backup job" in Live Library (https://livelibrary.osisoft.com).

You can re-run the migration tool by clicking **Tools** > **Notifications Migration** from PI System Explorer. The migration tool first determines and displays information on your state, which is one of:

- **Not migrated**

  Legacy notifications may be created and managed using the **Notifications** plug-in on the bottom left of the PI System Explorer window. You must use the 32-bit version of PI System Explorer to manage legacy notifications.

  You can choose to be in the "Not migrated" state by choosing the appropriate option while running the migration tool. In this state, you cannot create new notification rules.

- **Migrating**

  Legacy notifications are read-only; only new notification rules may be created. You can stop and start legacy notifications while in this state.

In the migrating state, you may run and test legacy notifications and new notification rules side-by-side to ensure that the migration is successful and that the new analyses and notification rules are working as intended. You can check the migration report, fix the errors indicated on legacy notifications, and then choose to re-run migration for only those legacy notifications that encountered errors in the previous migration run.

You may move to the migrated state when you have successfully tested that your migrated notification rules match your legacy notifications. Move to the migrated state only when the following are true:

- You intend to continue to work only with the new notification rules
- You will never need to revert to your legacy notifications
- You have successfully migrated all the legacy notifications that you wanted to migrate

- **Migrated**

  When you move to this state, you can create only new notification rules in the system. Your legacy notifications are deleted and you cannot revert to your legacy notifications or create legacy notifications.

  From the migrated state, you may re-run the migration tool to revert to the "Not migrated" state; however, your legacy notifications cannot be recovered.

Sample migration report

The migration tool generates a report when it completes the migration process. The report includes information on whether legacy notifications are migrated successfully. The "Errors" section and the "Not Supported" sections indicate notifications that are not migrated. The "Warnings" section indicates notifications with changed functionality, and the report provides more information.

> **Note:**
> Migration report can be found in `%userprofile%\Documents` folder of the person running the migration.

This is a sample migration report summary:

Notifications Migration Report for TMIPVDSHD01

Summary

Started: 8/24/2016 1:40:02 PM
Ended: 8/24/2016 1:59:07 PM
Notification templates: 26 of 32 migrated successfully
Notifications: 1736 of 1832 migrated successfully
Notifications or templates with errors: 92
Notifications or templates with warnings: 10
Notification Rules created: 4503
Analyses created: 4503
Legacy history instances processed: 0
Event frames created: 0
Warnings (2)
1. Acknowledgment Web Page hyperlink has been migrated to a PI Coresight event details hyperlink.
2. PI Coresight event details hyperlink requires a PI Coresight installation.

Notification Templates (32)

Outcome	Element Template	Legacy Notification Template	Migrated Notification Rule Template	Migrated Analysis Template	Messages
Not Migrated (5)					
There was an error migrating legacy notification templates. The notification templates below have not been migrated, please see the messages listed below for more information on how to resolve the issue.					
		P442_LIGNE_PANNE_COMM			Errors (1) 1. The notification had no target element.
		P442_TRANSFO_PANNE_COMM			Errors (1) 1. The notification had no target element.
		UM7650_PANNE_COMM			Errors (1) 1. The notification had no target element.
	\AADT_DEFAUT\ElementTemplates[SA_BAT]	SA_PTROSEE			Errors (1) 1. Monthy or weekly periodic time rule is not supported.
	\AADT_SUPERVISION\ElementTemplates[Interface PICybectecSMP-PROD/Frontal]	Etat_Interface_PI-Cybectec_SMP_ou_Frontal DPTNE			Errors (1) 1. The analysis is not convertible.
Migrated With Altered Configuration (1)					
The legacy notification templates were migrated to analysis templates and notification rule templates. During migration some changes were made to the migrated analysis templates and notification rule templates. No changes were made to the legacy notification templates. Please see the messages below for more information on the changes that were made.					

In addition to the summary, the report also contains information on legacy and migrated notification rules and templates, and specific information such as notification rules and analysis names, number of migrated event frames, and the number of processed history events.

Additional resources

Training video

For more information on using notifications in PI Asset Framework, watch the videos on the OSIsoft Learning channel playlist:

https://www.youtube.com/playlist?list=PLMcG1Hs2Jbcs43wQ2qO2dxW72PqGhZjKW&rel=0

Management of analyses and notification rules

To manage your analyses or notification rules, use the Management plug-in, accessible on the bottom left of the main screen of the PI System Explorer. The Management feature (which includes analyses and notification rules) is available if your installation of PI System Explorer includes the Management plug-in. You can view and manage the analyses or notification rules in the current PI AF database. In the **Management** pane, search for analyses or notifications, backfill or recalculate your analyses and enable or disable analyses and notification rules in bulk.

Topics in this section

- Search for analyses or notification rules
- Management of analyses
- Management of notification rules

Search for analyses or notification rules

Procedure

1. Navigate to the Management pane.
2. Select what you want to search: **Analyses** or **Notification Rules**.
3. Use the provided search or create your own.

To	Then
Use a provided search	Select **All**, **Enabled** or **Disabled**. These are view-only and cannot be modified, as indicated by the (**View selected search**) icon.

Create a custom search	**Note:** Customized search is only visible to the user who created it on the computer where it was created. Creating your own search is a new feature in PI AF 2017. Refer to the previous versions of PI System Explorer User Guide if you are using an earlier version of PI AF. a. Click the (**Add new search**) button. b. Select a search criteria. Choices are: ▪ Name ▪ Description ▪ Element Name ▪ Template ▪ Category ▪ Enabled/Disabled: enabled or disabled ▪ Service Status: PI Analysis/Notifications Service status ◦ Error ◦ Running ◦ Stopped ◦ Suspended ◦ Warning **Note:** **Service Status** is based on the PI Analysis Service connection and its running status. This cannot be combined with other search criteria.
View/edit a search	Click the (**View/edit selected search**) button to view or edit your search.
Delete a search	To delete, click the (**Delete selected search**) button.

Management of analyses

The **Management** plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Analyses** button to view and manage your analyses.

Note:
For information on notifications management, see Management of notification rules.

Analyses pane

The **Analyses** pane displays information about selected analyses in these columns:

- Status
- (Automatic recalculation)

- (Analysis Type)
- Element
- Name (of analysis)
- Template
- Backfilling (status)

> **Note:**
> In the **Analyses** pane, you may see the number of analyses depicted with the "~" character, or with the word "about"; for example: "~1000" or "1-500 of about 1000". In some cases, the exact number of analyses is not known until they are all loaded.

- **Analysis selection**

  With PI Server 2015 R2 and later versions, analyses results are displayed in pages of 500 results each. You can select:

  - Specific analyses by clicking on each individual analysis check box.
  - All 500 results on a page by clicking the checkbox on the top-left corner.
  - (When more than 500 analyses exist.) All analyses on all pages by clicking **Select all analyses on all pages**.

    > **Note:**
    > Clearing specific analyses automatically makes your subsequent operation page-based. For example, if you first select all 1000 analyses, and then clear two, the operation will be performed on 498 (500 minus 2) and not 998 (1000-2) analyses.

- **Operations**

  You can enable, disable, backfill/recalculate the analyses you have selected. For specific selection criteria, see the **Analysis selection** section above. From PI AF 2017 R2 and later, you can enable or disable automatic recalculation of the analyses you selected. For more information, see Enable or disable automatic recalculation for multiple analyses.

  In order to backfill/recalculate, you need Execute permission on the analyses. Proper permission can be obtained by mapping your account to `Asset Analytics Recalculation` identity. Note that manual recalculation is only available for PI Analysis Service 2016 R2 or later. In addition, PI Data Archive that stores PI points must be running version 2016 R2 or later. For more information, see Backfill or recalculate data for multiple analyses.

  Expression, rollup, or SQC analyses can backfill or recalculate data if the analysis outputs are mapped to PI point attributes. If PI points already have data for the backfill time period, you can retain existing data and fill in missing data, or delete the data and recalculate.

  Event frame generation analyses can also recalculate event frames over a specified time period. The recalculation process, regardless of the selected option, automatically deletes all existing event frames for that time period, as well as annotations on affected event frames.

- **Pending Operations**

  See the status of following bulk operations: enable, disable, backfill or recalculate a group of analyses.

- **Analysis details**

  Select an analysis to view detailed information.

  - To see a summary of the analysis configuration and status, click the **Overview** tab.
  - To see error details, click the **Errors and Warnings** tab.
  - To view or edit the selected analysis, click **Analysis Configuration** to go to the **Analyses** tab of its associated element.

Management of notification rules

The **Management** plug-in gives you a choice of two radio buttons to manage either your analyses or your notification rules. Select the **Notification Rules** button to view and manage your notification rules.

Notification Rules pane

The **Notification Rules** pane displays information about selected notifications rules in these columns:

- Status
- Element
- Notification rule name
- Template (Notification rule template)
- Categories

You can select or deselect notification rules from the list of configured notification rules, and monitor individual status on the Notification Rule Details pane; you can also enable or disable selected notification rules from the Operations pane.

Operations, Pending Operations, and Notification Rule Details panes

The other screens on the **Management** plug-in include:

- An **Operations** pane where you can enable or disable notification rules.
- A **Pending Operations** pane that provides information on whether an operation is queued or complete.
- A **Notification Rule Details** pane that is displayed when you select a notification rule, and has details, status and error information about the notification rule. Click on **Notification rule configuration** to go back to the element **Notification Rules** tab.

Process models in PI AF

A PI AF model consists of connected elements that represent a logical model of your process. A model is itself an element, but with two additional element properties: layers and connections. Elements in the model can represent physical entities in your process, such as tanks, pipes, and process units, or logical entities, such as recipes and summary data. The model is composed of connected elements.

PI AF models can be simple, containing only a handful of elements, or they can be very complex, containing thousands of elements and measurements. The size of the model is only limited by the data available to fulfill the information requirements of an analysis on the model.

Topics in this section

- The scope of a PI AF model
- Guidelines for PI AF models
- Submodels
- Element types in models
- Create PI AF models
- Edit PI AF models
- Ports and connections
- Layers

The scope of a PI AF model

While the number or variety of elements that a model encompasses does not change how a model is stored, it does help in planning both the initial model design and the information needed to complete a model analysis and get meaningful results. When considering the scope of a model, remember that it needs to be small enough for the relationships between properties to be well defined, but large enough to include some redundant data. Redundant data can be calculated from other data in the model.

At different levels of scale, or modeling scope, the consumers of the resulting information will change. For example, an engineer looking at equipment performance of a section of the facility may need more detailed information than a resource planning individual who tracks materials throughout a facility. The planner needs measurements in the model that are different from the measurement needs of the engineer. Elements that are common to both models use the same source elements; this aids in the construction of a variety of models using the same element library.

Three modeling levels, in order of increasing scope, are the following:

- Unit Model
  - This level of modeling typically includes the smallest details of information within a processing unit or area. This scope is useful for monitoring equipment performance and is used primarily by the engineers on that equipment. Auxiliary loops and heating

systems that occur at this modeling level might not have influence outside of this area, and therefore would not be included at the next scale.

  - To perform a meaningful analysis on a unit level model, detailed data must be available on the materials and quantities within the unit. If there is only information available on the parameter of the unit (the inputs and the outputs), this is an indication that the data model is at its smallest granularity.

- Multi-Unit Model
  - When considering unit-to-unit type models, detail within the unit is typically summarized by the connections to and from that unit. Material added and sent to storage areas (tanks, stockpiles, etc.) would also be included.
  - At the multi-unit modeling level, measurements within the unit that do not affect the main inputs and output quantities are not included in the model.
- Boundary Model (Facility and Business Unit)
  - It is useful to create an analysis for the materials entering and leaving the "fence line" of the facility, further summarizing the unit-to-unit information to only the transactions to and from the process from material transfer points (shipping docks, weigh scales, tank feeds, pipelines). This level is also useful to analyze transactions between business units within a facility.

Guidelines for PI AF models

Consider the following rules as you set up a model for analysis:

- Use a systematic approach to identify the location and calibration setup for each instrument in the model.
- Associate every instrument with the correct flow, and make sure that temperature and specific gravity corrections are applied to flow measurements if needed. Check whether this function of correction is performed by your control system or historian.
- Validate tank/inventory properties by building a tank-only model, including shipments and receipts. Enter the materials list with its properties at the same time.
- Determine the calibration tolerance of each instrument (you can use ISO 5167 for orifice meters), or refer to meter data sheets and manufacturers.
- Build and examine individual unit models before attempting to connect them and run a multi-unit analysis.
- After you achieve full unit connectivity, examine the entire drawing, and then run a balance analysis to identify flows and connections that cannot be solved.

Submodels

A model is also an element, which means that a model can be composed of other models - referred to as submodels. This allows for either top-down or bottom-up development of a plant model. The boundary elements of a model normally define the elements with ports that can be used for connections outside the model.

Element types in models

Six core element types are used to enforce connectivity rules between elements of a model.

- Use a **Node** to represent a physical entity in your model, such as a tank, valve, or process unit.
- Use **Measurement** to indicate that the element is used for ascertaining dimensions, quantities, capacities, etc., such as, meters or scales.
- Use **Flow** to indicate that the element carries material from one element to another.
- Use **Transfer** as a temporal flow. The existence of a transfer in a model is only available in the context of time, for example, in a case. Transfers can also be accessed by performing time-based searches of the database.
- Use **Boundary** to define the input and output ports for the model.
- Use **Other** to represent a logical collection of attributes, such as a recipe.

In addition to the core element types, two other types are supported: **Any** and **None**. Use the **Any** and **None** element types when you define the connectivity rules for a port.

Create PI AF models

You create a PI AF model in the **Elements** browser.

Procedure

1. In the navigator, click **Elements**.
2. Right-click the collection of elements and click **New Model**.
3. In the Choose Model Template window, select a template on which to base the model or select **None**.
4. Click **OK**. The model configuration tabs appear in the viewer.
5. In the **Name** field of the **General** tab, enter a name for the model.
6. Optional. In the **Description** field, enter a description for the model.
7. The read-only **Template** and **Type** fields list the template and element type chosen when the model was created. To view the template properties, click . Element types are described in Element types in models.
8. Optional. You can organize objects by grouping them into categories. To browse the available categories, click .
9. To assign a default attribute, select the attribute from the **Default Attribute** drop-down list. Note that you must add attributes in the **Attributes** tab first before they appear in the list. This field is read only if the model is based on a template; the box displays the attribute specified in the template.
10. Optional. To specify location attribute traits for the model, click the **Location** link. For more information, see Set location attribute traits.

11. Optional. To specify asset health attribute traits for the model, click the **Health** link. For more information, see Set health attribute traits.
12. Optional. To configure access permissions for the new model that are different from those inherited from the Elements collection, click the **Security** link. For more information, see Configure security for objects.
13. Click **OK** and check in your work.

Edit PI AF models

You view or edit PI AF models in the <Model Name> Connections window.

Procedure

1. Right-click on a model in the **Elements** browser and click **Model Connections**.

   The <Model Name> Connections window shows a visual representation of the elements in the model and how they are connected.

2. To edit the model:
   - Right-click an element in the **Connections** pane to create a new connection or to view or edit the properties of the element.
   - Click an element to center it in the **Connections** pane and show any other connected elements. You can move along the flow of the model this way, element by element.
   - Click a connection to view or edit the connection or to add another source or destination.
   - Right-click a connection to make another connection, view the properties of the selected connection, or to copy and paste the properties of the connection.

Ports and connections

Elements in a model are connected through any number of ports, which are defined by the element template. A port can be defined as an input port, an output port, or as an undirected port. The port defines how many connections can be made and the types of elements that can be connected.

In analyses, directed ports (input and output ports) represent positive material flow and are used by connections. Undirected ports are used by attachments of meters and analyzers. The most common type of attachment in a model is a measurement or meter attached to a flow element.

A connection represents the link between the ports of two elements. The ports, which are defined by the element template, can be defined as input ports, output ports, or undirected ports.

Topics in this section

- Create ports
- Create connections

Create ports

Before you start

To specify a port as the default port, you must open the Properties window.

Procedure

1. In the Elements browser, select an element and then click the **Ports** tab in the viewer.

   > **Note:**
   > If the element is based on a template, you cannot add a port unless the template allows extensions.

2. Click **New Port**, and create a port for the element. Right-click an existing port to view or edit its properties.
3. Configure the port.
   - **Port Type:** Select the port type: **Input**, **Output**, or **Undirected** (for meters, for example).
   - **Allowed Categories:** Select the categories of which the port is allowed to be a member.
   - **Maximum Connections:** Specify the maximum number of connections that can be made to the port. Enter zero for an unlimited number of connections.
   - **Connection Type:** Select the type of element to which the port can connect, for example, Node, Boundary, Measurement, and so on.
   - **Allowed Templates:** Select elements allowed to connect to the port. Select only elements created from the selected template.

Create connections

Use the **Connections** tab to display the connections in a model.

Procedure

1. To create a new model in the Elements browser, right-click the collection of elements as described in Create PI AF models, and click **New Model**.
2. Follow these steps to fill in the **Connections** tab.
   a. Right-click in the field and select **New Connection**.
   b. In the Make Connection window, right-click an existing connection to view or edit its properties.
   c. In the **Source** field, select the source for the connection. Select the appropriate port of the source in the corresponding **Port** field.
   d. In the **Destination** field, select the destination for the connection. Select the appropriate port of the destination in the corresponding **Port** field.

   To include child elements, select the **Include Child Elements** check box.

Layers

You can organize the elements of a model into layers. Layers provide a mechanism for including or excluding portions of the model as needed for analysis. An element of a model can belong to

more than one layer. You can also use the Layers feature with a graphic modeling tool, such as PI ProcessBook, to provide a visual overlay functionality.

Create layers

Create a Layer to enable you to include or exclude portions of a model.

Procedure

1. In the Elements browser, select the model to which you want to add a layer.
2. Click the **Layers** tab in the viewer.
3. Right-click and click **New Layer**.
4. In the **Name** field of the Layer Properties window, enter a unique name for the layer.
5. Optional. In the **Description** field, describe the purpose of the layer.
6. In the element list, select one or more elements that comprise the layer.
7. Click **OK**.

PI AF utilities and plug-ins

After you install PI AF, several utilities are available to assist you in your administration of PI AF and management of plug-ins.

Topics in this section

- Launch PSE with command line options
- PI AF Diagnostics utility
- AF Update Plug-in Configurations utility
- Set PI AF server utility
- Capture AF SDK event trace output
- Track PI AF changes with Audit Trail
- View installed plug-ins
- Command-line plug-in registration
- Create an XML registration file
- Create an SQL registration script
- Register plug-ins with generated SQL scripts
- Plug-in provider management

Launch PSE with command line options

The PI System Explorer (PSE) can be invoked with command line options that control its initial selection. The PI System Explorer application is named **`AFExplorer.exe`** and is located in the `\PIPC\AF` folder.

Procedure

1. Open a Windows command window and change to the `\PIPC\AF` folder.
2. Type:
   ```
   afexplorer parameter=paramValue
   ```
   where *`parameter`* is one of the following three parameters:
   - **`/system`**
   - **`/database`**
   - **`/navigator`**

   To display a list of available parameters, type:
   ```
   afexplorer /?
   ```

AFExplorer parameters

The following table lists the available parameters for the **`AFExplorer`** utility.

Parameter	Description	Example
/system	Sets the system parameter to the hostname of the PI AF server to which PSE should connect by default.	`afexplorer /system=MyAFServer`
/database	Sets the database parameter to the name of the PI AF database that PSE should open initially.	`afexplorer /database=MyAFDatabase`
/navigator	Sets the navigator parameter to the browser plug-in that should be selected initially in PSE.	`afexplorer /navigator=Elements`

Navigator parameter values

The following table lists the available values for the **navigator** parameter.

Parameter value	Example
`Elements`	`afexplorer /Navigator=Elements`
`EventFrames`	`afexplorer /Navigator=EventFrames`
`Library`	`afexplorer /Navigator=Library`
`UnitOfMeasure`	`afexplorer /Navigator=UnitOfMeasure`
`Analyses`	`afexplorer /Navigator=Analyses`
`MyPI`	`afexplorer /Navigator=MyPI` **Note:** Use only if legacy notifications are present on a PI AF client computer.
`Notifications`	`afexplorer /Navigator=Notifications` **Note:** Use only if legacy notifications are present on a PI AF client computer.
`AFContactNavigator`	`afexplorer /Navigator=AFContactNavigator`
`Management`	`afexplorer /Navigator=Management`

PI AF Diagnostics utility

The PI AF Diagnostics (**AFDiag**) utility is a command-line utility that you can use to enable or disable PI AF server features and perform other administrative functions. The utility makes a direct connection with the associated SQL Server database and requires the SQL Server `sysadmin` or `db_AFadmin` role.

AFDiag is located in the `\PIPC\AF` folder.

Topics in this section

- Grant permissions for PI AF Diagnostics utility
- Run the AFDiag utility
- AFDiag utility parameters
- Audit Trail implementation

Grant permissions for PI AF Diagnostics utility

To use the PI AF Diagnostics utility, you need to grant the `db_AFadmin` database role to the SQL Server Login.

Procedure

1. In the Microsoft SQL Server Management Studio, connect to the SQL Server instance in which the PIFD database resides.
2. Under the SQL Server instance, expand the **Security** folder; then expand the **Logins** folder.
3. Right-click the login that corresponds to the appropriate Windows user and select **Properties**.
4. Select the **User Mapping** page.
5. Select the row for the PIFD database.
6. Select the **Map** check box for the PIFD database.
7. With the database still selected, select the **db_AFadmin** database role check box, as shown in the following figure.

8. Click **OK** to save your changes and close the Microsoft SQL Server Management Studio.

Run the AFDiag utility

You run the **`AFDiag`** utility in an elevated Windows command prompt window.

Procedure

1. As an administrator, open a Windows command prompt and change directory to `\PIPC\AF`.
2. Choose one of the following actions.

To ...	Do this ...
Display current configuration settings	Type `afdiag` with no parameter specified. A list of current settings is displayed for the PI AF server, database, Active Directory, and other configuration items such as audit trail, file extensions, and maximum size for files attached to PI AF objects.
Display a list of available parameters	Type `afdiag /?`

Execute a specific configuration task	Type `afdiag parameter` where *parameter* is one of the parameters listed in AFDiag utility parameters.

AFDiag utility parameters

Parameters for the **AFDiag** utility are listed in alphabetical order below.

- **/ActiveDirectory:*string* or /AD:*string***

  Tests access to Active Directory using the currently configured settings in the PI AF server. An optional user account can be specified to test another account. If an account is specified or set in the PI AF server settings, you also need to specify the **Password** parameter.

- **/AddIdentity:*string* or /AI:*string***

  Adds a security identity with the specified permissions to all security strings in the system. For example, to allow read and write access for *IdentityName* to all security strings, enter:
  `/AI:"IdentityName:A(r,w)"`

  If the specified security identity does not exist, it is created. Use PI System Explorer to create and/or configure mappings to the new identity.

- **/AuditTrailCleanupAdd:*string* or /ATCA:*string***

  Adds the specified user to the audit trail cleanup list. The audit cleanup job deletes audit records associated with this user from the PI AF SQL Server database.This parameter may be specified more than once.

- **/AuditTrailCleanupRemove:*string* or /ATCR:*string***

  Removes the specified user from the audit trail cleanup list. The audit cleanup job no longer deletes audit records associated with this user from the PI AF SQL Server database.This parameter may be specified more than once.

- **/CertificateAdd or /CA**

  Adds the specified client certificate to the PI AF SQL Server database. Use the **Password** parameter to specify a password for the certificate, if required.

- **/CertificateList or /CL**

  Lists the client certificates stored in the PI AF SQL Server database.

- **/CertificateRemove or /CR**

  Removes a client certificate from the PI AF SQL Server database by specifying the name of the certificate.

- **/CertificateSet:*string* or /CS:*string***

  Sets the server certificate in the PI AF server configuration file to the specified file. Use the **Password** parameter to specify a password for the certificate, if required.

- **/ChangeID:*string* or /CID:*string***

  Changes the ID for the PI AF server to the specified GUID.

- A return value of 1 means that the configuration change will be delayed.
- A return value of 2 means that you must restart the PI AF server.
- A negative return value indicates an error has occurred.

- **`/ClearChangeTables or /CCT`**

  Clears the **`findChanges`** and **`afdiag`** tables, which record information on changes to the system.

- **`/DeleteAuditTrail or /ATD`**

  Disables audit trail feature and deletes audit trail records in the PI AF SQL Server database. This operation permanently deletes the audit trail records, and requires `sysadmin` privileges. The audit trail records cannot be recovered if a delete is performed.

- **`/DeleteCases:`*`string`*` or /DelC:`*`string`***

  Deletes cases from the PI AF SQL Server database with a start time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

  - To specify a time, include a time string in **`hh:mm:ss`** format.
  - To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
  - To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

  For example, to delete cases that start between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelC:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

- **`/DeleteEventFrames:`*`string`*` or /DelEF:`*`string`***

  Deletes event frames from the PI AF SQL Server database with an end time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

  - To specify a time, include a time string in **`hh:mm:ss`** format.
  - To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
  - To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

  For example, to delete event frames that end between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelEF:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

  > **Note:**
  > Child event frames are also deleted, even if they do not match the criteria.

- **`/DeleteTransfers:`*`string`*` or /DelTR:`*`string`***

  Deletes transfers from the PI AF SQL Server database with an end time between the dates specified in local time format, using standard date notation or PI time syntax (described in PI time). If only one time is provided, it is treated as the end time.

- To specify a time, include a time string in **`hh:mm:ss`** format.
- To specify a database, include a database string in **`/Database:"`***DBname***`"`** or **`/DB:"`***DBname***`"`** format.
- To specify a template in a database, include a template string in **`/Template:"`***Template Name***`"`** or **`/Temp:"`***Template Name***`"`**format, in addition to the database string.

For example, to delete transfers that end between 11 P.M. on January 31st, 2017 and 4 P.M. on June 21st, 2017, enter `/DelTR:"2017-01-31 23:00:00";"2017-06-21 16:00:00"`.

- **`/EnableAuditTrail or /AT`**

  Enables audit trail feature for the PI AF SQL Server database.

- **`/EnableAuditTrailCleanup[-] or /ATC[-]`**

  Enables audit trail cleanup feature for the PI AF SQL Server database. Requires `sysadmin` privileges.

- **`/EnableExternalDataTables[-] or /DT[-]`**

  Enables support for external PI AF tables.

- **`/EnablePropagationOfTargetDeletion[-] or /PTD[-]`**

  Enables support for propagating the deletion of targets (elements) to the referencing analyses and notifications.

- **`/ExeFile:`*`string`*` or /F:`*`string`***

  The path to the PI AF server executable file. Default value is `AFService.exe`.

- **`/ExternalDataTablesAllowNonImpersonatedUsers[-] or /DTImp[-]`**

  Enables support for external PI AF tables for non-impersonated users.

- ***`@file`***

  Reads response file for more options. The response file must contain one parameter per line. Comment lines start with the '#' character.

- **`/FileExtensions:`*`string`*` or /FE:`*`string`***

  Defines the types of file objects that can be attached to a PI AF file object, such as an annotation. The following file types are supported:

  - MS Office: csv, docx, pdf, xlsx
  - Text: rtf, txt
  - Image: gif, jpeg, jpg, png, svg, tiff
  - ProcessBook: pdi

  Enter the extensions as a colon-separated list, for example:

  ```
  /FE:docx:xlsx:csv:pdfd:pdi:jpg:png
  ```

- **`/FileExtensionsAdd:`*`string`*` or /FEA:`*`string`***

  Adds an additional file extension to the list of allowed PI AF file object types. See the list of supported file types above.

You can specify this parameter more than once.

- **`/FileExtensionsRemove:`*`string`*` or /FER:`*`string`***

  Removes a file extension from the list of allowed PI AF file object types. See the list of supported file types above.

  You can specify this parameter more than once.

- **`/FileMaxLength:`*`integer`*` or /FML:`*`integer`***

  Defines the maximum size in megabytes of a PI AF file object, such as an annotation. A value of zero disables support for all files. The default maximum allowed file size is 10MB.

- **`/FindDuplicates or /FD`**

  Searches for duplicate entries in all object tables and returns a list of tables. It identifies how many duplicates were found in each object table and logs them in an XML file. OSIsoft recommends that you make a backup of the PI AF SQL Server database (PIFD) before you proceed to delete the duplicate entries.

- **`/NewID or /NID`**

  Generates a new ID for the PI AF server.

  - A return value of 1 means that the configuration change will be delayed.
  - A return value of 2 means that you must restart the PI AF server.
  - A negative return value indicates an error has occurred.

- **`/Password or /PWD`**

  Specifies a certificate password for the `ActiveDirectory`, `CertificateAdd`, `CertificateSet`, or `PasswordEncryptionCertificate` options.

- **`/PasswordEncryptionCertificate:`*`string`*` or /PEC:`*`string`***

  Replaces the default password encryption for external database passwords with a custom certificate file. Use the **`Password`** option to specify a password for the certificate, if required.

- **`/PlugInVerifyLevel:`*`level`*` or /VL:`*`level`***

  Configures the level of verification required for plug-ins to run. Valid levels are:

  - **`None`**

    Disables validation; runs all plug-ins.

  - **`AllowUnsigned`**

    Runs unsigned plug-ins and plug-ins with valid signatures.

  - **`RequireSigned`**

    Runs only plug-ins with valid signatures.

  - **`RequireSignedTrustedProvider`**

    Runs only plug-ins with a valid signature from a trusted provider.

- **`/Port:integer or /P:integer`**

  Tests the specified port to the PI AF server. This is used to perform a basic port test to see if specified port on the computer used by the PI AF server can be opened and something is listening for a connection on the port. The standard ports used by the PI AF server are 5457 and 5459. This is similar to attempting to test a port using Telnet. It tests all IP addresses for the computer (both IPv4 and IPv6 addresses). Typically, you would test both ports 5457 and 5459.

- **`/RebuildPathCache or /RPC`**

  Rebuilds the path cache to each element in the PI AF SQL Server database. The path cache can become outdated after significant data insertions, edits, and/or deletions. Rebuilding the path cache can improve the PI AF server's performance.

  > **Note:**
  > The stored procedure used to rebuild the path cache is not supported on a secondary PI AF server (a subscriber) in a PI AF collective.

- **`/Reindex or /RI`**

  Completely rebuilds every index in the PI AF SQL Server database. This substantially improves the PI AF server's performance after a massive data insertion.

- **`/ResetAdministrator:string or /SRA:string`**

  Resets the permissions on the `Administrators` identity to allow full privileges on the local PI System. This parameter requires `sysadmin` or `db_owner` privileges.

  The default `/sra` enables privileges for the `BUILTIN\Administrators` account.

  `/sra:domainName\username` enables privileges for a particular user account.

- **`/Silent[-] or /S[-]`**

  Runs silent mode and prevents message display.

- **`/TrustedProviderAdd:providername or /PA:providername`**

  Adds the specified provider to the list of trusted plug-in providers. To add providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/TrustedProviderList or /PL`**

  Displays a list of trusted plug-in providers. To list providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/TrustedProviderRemove:providername or /PR:providername`**

  Removes the specified provider from the list of trusted plug-in providers. To delete providers, you must have SQL Server `sysadmin` server role or `db_AFadmin` database role.

- **`/UomCaseSensitive[-] or /UCS[-]`**

  Changes configuration of UOM abbreviations from case insensitive to case sensitive in the PI AF SQL Server database. When enabled, only a 2015 R2 (v2.7.5) or later PI AF Client can connect.

  For more information, see Configuration of case-sensitive UOM abbreviations.

- **`/UpgradeAuditTrail or /ATU`**

  Upgrades audit trail records in the PI AF SQL Server database. Supports upgrades from PI AF 2.6 or later.

- **`/Version or /V`**

  Displays version information.

Audit Trail implementation

You can use the audit trail feature to examine audit trail records in the PI AF SQL database. Once enabled, audit trail records are created using SQL Server Agent jobs. You can view the audit trail with Process Explorer.

Enable audit trail

To enable the audit trail feature, you use the **`AFDiag`** utility and the **`EnableAuditTrail`** (**`/AT`**) parameter. For example:

```
afdiag /AT
```

Disable audit trail

To disable the audit trail feature and delete all audit trail records permanently, you must have `sysadmin` privileges on the SQL Server. To disable the audit trail feature, you use the **`AFDiag`** utility and the **`DeleteAuditTrail`** (**`/ATD`**) parameter. For example:

```
afdiag /ATD
```

Once you use this parameter, the audit trail is not recoverable.

Requirements

The audit trail feature uses SQL Server Change Data Capture (CDC) to generate an audit trail. CDC is supported in the following versions of SQL Server:

Version	Standard	Enterprise	Developer [1]	Evaluation [1]
SQL Server 2008	✗	✓	✓	✓
SQL Server 2008 R2	✗	✓	✓	✓
SQL Server 2012	✗	✓	✓	✓
SQL Server 2014	✗	✓	✓	✓
SQL Server 2016 [2]	✗	✓	✓	✓
SQL Server 2016 SP1 and later [2]	✓	✓	✓	✓

[1] Development system only.

[2] Only supported in PI AF 2016 and later. For more details on CDC support, see the OSIsoft Tech Support article KB01410 - PI AF error: Could not find stored procedure sys.sp_cdc_parse_captured_column_list (https://techsupport.osisoft.com/Troubleshooting/KB/KB01410).

The audit trail feature has the following additional requirements:

- The SQL Server Agent must be running before you enable the audit trail feature.
- You must be a member of the sysadmin role on the SQL Server that contains the PI AF SQL database.

Previous versions

The audit trail feature that was released in PI AF 2.6.0 or later is supported by this installation and is upgraded when the PI AF SQL scripts are executed. At that time, the audit trail records are upgraded to the current format. If for some reason a failure occurs during the upgrade, you can use the **AFDiag UpgradeAuditTrail** (**/ATU**) parameter to fix the audit trail tables and records.

Versions of the audit trail feature prior to PI AF 2.6.0 have a different format and are no longer updated with new change records after an upgrade of the PI AF SQL database. Existing tables and data remain intact, but new records cannot be added.

Audit trail support in PI AF collectives

The audit trail feature is not supported on secondary members of PI AF collectives. If the audit trail feature is currently enabled on a server that you wish to add as a secondary server to a PI AF collective, you must run the **AFDiag** utility and disable the feature. Audit trail data is only stored on the primary member of a PI AF collective and is not replicated to any secondary member of a PI AF collective.

Note:
When you are designing SQL Server backup procedures for your PI AF data, please be aware that if the primary member of a PI AF collective becomes unavailable, the audit trail data will also be unavailable. If the primary member of a PI AF collective cannot be recovered from a backup, the audit trail data cannot be recovered either.

Re-enabling audit trail

If you add an existing PI AF server with audit trail enabled to a collective as a secondary member, but later remove that server from the collective, you need to re-enable the audit trail feature on that particular server. Otherwise, auditing remains disabled.

AF Update Plug-in Configurations utility

The **AFUpdatePlugInConfigurations** utility provides a **Repair**, a **CreateConfig**, and a **ReplacePIServer** feature that enable you to perform bulk updates of attribute configuration strings with a single command. After you run the utility, click Refresh in PI System Explorer to see the changes.

The following table lists the available parameters for the **AFUpdatePlugInConfigurations** utility. Only one feature can be specified at a time in combination with the **/Root**, **/List**, and **/EventFrames** parameters.

Parameter	Description
/Root:*string*	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. Specifies the PI AF server or database on which to operate. Enclose the entire parameter string in quotation marks (" ").

Parameter	Description
/Repair[-]	Default parameter if no parameter is specified. For attributes on the PI AF server or database specified in the **/Root** parameter, corrects PI Point data reference attributes for which the stored configuration string has become out of synchronization with the PI Data Archive. This can occur for the following reasons: • Deleted PI points: When PI points are deleted and then recreated with the same name, the ID of the new point does not match the ID in the stored configuration string. • Renamed PI points: When PI points are renamed, the stored configuration string still uses the old PI point name. • Unresolved attributes: If the PI point to which a data reference points is not yet created, the stored configuration string does not contain the point ID.
/CreateConfig [-]	For attributes on the PI AF server or database specified in the **/Root** parameter, creates the PI point if it does not already exist, or updates it with any changes. This is the same operation as when you right-click an element or attribute in PI System Explorer and choose **Create or Update PI Point**.
/ReplacePIServer: *string*	For PI Point data reference attributes on the PI AF server or database specified in the **/Root** parameter, redirects attributes to a different PI Data Archive. Use a colon (**:**) to precede the existing PI Data Archive name. Separate the existing PI Data Archive name from the new PI Data Archive name with a semi colon (**;**) . Short form: **/Replace**
/List [-]	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. For the PI AF server or database specified in the **/Root** parameter, lists all attributes to be operated on.
/EventFrames:*string*	Use in conjunction with **/Repair**, **/CreateConfig**, and **/Replace** parameters. Performs specified operation on each attribute of all event frame templates and event frames that occurred between the start and end time specified, format, using standard international date notation YYYY-MMDD. Use a semi colon to separate the start and end time. Short form: **/EF**
/? or **/help**	Displays list of parameters
@*file*	Uses the specified file to provide additional input arguments. The file should contain one argument per line. Comment lines start with the '**#**' character.

/Repair syntax example

To repair stored configuration strings in the specified PI AF database so that they correctly map to the PI points on the PI Data Archive, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /Repair
```

/CreateConfig syntax example

To perform the CreateConfig operation in bulk on all attributes in a PI AF database, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /CreateConfig
```

To perform the CreateConfig operation on all attributes on a specific PI AF server, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer" /CreateConfig
```

/ReplacePIServer syntax example

To redirect all PI point data reference attributes in a specified PI AF database to a new PI Data Archive, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase"
   /ReplacePIServer:OldPIDataArchive;NewPIDataArchive
```

/EventFrames syntax example

To perform the CreateConfig operation in bulk on all event frame attributes in a PI AF database over a specific time range, use the following syntax:

```
afupdatepluginconfigurations "/Root:\\MyAFServer\MyAFDatabase" /EF:"2017-01-31
23:00:00";"2017-06-21 16:00:00"  /CreateConfig
```

Set PI AF server utility

The SetPISystem utility (**setpisystem**) enables you to configure known PI AF servers. **setpisystem** is located in the PIPC\AF directory.

SetPISystem utility parameters

The following table lists the available parameters for the SetPISystem utility.

Parameter	Short form	Description
/Name:*string*	**/N**	Specifies the name of the PI AF server to modify or create. If not specified, the default PI AF server is used.
/Host:*string*	**/H**	Specifies the hostname for the PI AF server.
/Protocol:*Tcp*|*NamedPipe*	**/C**	Specifies the protocol for the PI AF server.
/Port:*integer*	**/P**	Specifies the port for the PI AF server.
/Timeout:*integer*	**/T**	Specifies the timeout for the PI AF server in seconds.
/AccountName:*string*	**/A**	Specifies the account name for the PI AF server.
/DefaultPISystem	**/D**	Sets the specified PI AF server as the default PI AF server.
/Remove	**/R**	Removes the specified PI AF server from the list of known PI AF servers.
/Silent[-]	**/S**	Establishes silent mode, which prevents message display.
/List	**/L**	Lists the current known PI AF servers.
/AddAlias:*string*	**/AA**	Adds the specified alias to the PI AF server.
/RemoveAlias:*string*	**/RA**	Removes the specified alias from the PI AF server.

Capture AF SDK event trace output

You use the **AFGetTrace** utility (**afgettrace.exe**) to capture event trace output from the AF SDK. Event tracing can help you debug an application and perform capacity and performance analysis.

Procedure

1. Open a command window and change directory to PIPC\AF.
2. Choose from the following actions:

To ...	Do this ...
Display syntax and parameters	At the command prompt, type: `afgettrace /?`
Run **AFGetTrace** with default settings	At the command prompt, type: `afgettrace` Default output goes to standard output.
Run **AFGetTrace** with specific parameters	At the command prompt, type: `afgettrace /parameter` Refer to AFGetTrace utility parameters for details on the parameters you can use.
Terminate event tracing	In the command window, type: `X` **Note:** If you close the command window without terminating **afgettrace**, trace events continue to be generated, which can slow down your AF SDK applications.

AFGetTrace utility parameters

The following table lists available parameters for the **AFGetTrace** utility.

Parameter	Short form	Description
/Provider:*string*	**/P**	Specifies the name of the event tracing session. You only need to specify to use an existing event trace provider. The default value is AFGetTrace.
/Level:{Critical|Error|Warning|Warning|Information|Verbose|Detail}	**/L**	Specifies the level of detail to be included in the events written by the AF SDK. Detail at or above severity of the level chosen is generated. The default value is Verbose.
/Keywords:{None|Server|Connection|Cache|Events|Trace|Data|All}	**/K**	Specifies the keywords used to determine the category of events that you want the AF SDK to write. The AF SDK writes the event if any of the event's keywords match the keywords specified in this setting. The default value is All.

Parameter	Short form	Description
/EnableAF[-]	**/AF**	Enables messages from the AF SDK message provider. The default value is `True`.
/EnablePI[-]	**/PI**	Enables messages relating to communication with the PI Data Archive from the MDA message provider. The default value is `True`.
/LogFile:*string*	**/Log**	Specifies the name of the log file for trace output messages. Messages are still displayed on standard output unless you specify the **Silent** parameter.
/LogFileMaxSize:*double*	**/FileMax**	Specifies the maximum size of the log file in megabytes. The trace ends once the size is reached. Enter `0` for no limit. The default value is `0`.
/Timeout:*integer*	**/TO**	Specifies the number of minutes before the trace ends. Enter `0` for no timeout. The default value is `0`.
/Mask:*string*	**/M**	Prevents any message containing the specified mask from being displayed. Enclose the mask in quotes if it contains spaces. This parameter can be specified more than once, but values must be unique.
/MaskPID:*UInt32*	**/MPID**	Prevents any message associated with the specified Process ID from being displayed. This parameter can be specified more than once. **Note:** This parameter is ignored if the **ProcessID** parameter is specified.
/MaskTID:*UInt32*	**/MTID**	Prevents any message associated with the specified Thread ID from being displayed. This parameter can be specified more than once. **Note:** This parameter is ignored if the **ThreadID** parameter is specified.
/NoHeader[-]	**/NH**	Disables header information on each message. Header information includes the time stamp, process identifier, and thread identifier.
/ProcessID:*UInt32*	**/PID**	Displays any message associated with the specified process ID. This parameter can be specified more than once.
/ThreadID:*UInt32*	**/TID**	Displays any message associated with the specified thread ID. This parameter can be specified more than once.
/WordWrap	**/W**	Word wraps output messages to the width of the console.
/Silent[-]	**/S**	Establishes silent mode, which prevents message display.

Track PI AF changes with Audit Trail

If Audit Trail is enabled for your system, users with administration privileges can use the utility to view changes to PI AF objects from all PI AF databases, or to a single object.

Beginning with PI AF 2017 R2, users with administration privileges can right-click an object in the browser or an object on a list in the viewer and click **Audit Trail Events** to review audit data specific to that object only.

See Audit Trail implementation for details about enabling the Audit Trail feature.

Each row in the table of the AF Audit Trail window contains data that identifies a specific change to a PI AF object. You can double-click a row to view details about that change in the AF Audit Trail Details window.

You can press CTRL+C and CTRL+V to copy and paste rows from either window into another document, such as a spreadsheet.

Review changes with the Audit Trail utility

Use the Audit Trail utility to track changes to one or more PI AF objects.

Before you start

- Only users with administrator privileges can use the Audit Trail utility.
- Both the PI AF client and PI AF server should be located in the same time zone for audit trail queries to work properly.

Procedure

1. Choose from the following actions:

To ...	Do this ...
Review changes to PI AF objects in all databases	Click **Tools** > **Audit Trail**.
Review changes to a single PI AFobject	a. Open the database that contains the object you want to audit. b. In the navigator, click the object type you want to audit. For example, to audit a table, click **Library**. c. In the browser, expand the tree until the object you want to audit is displayed in either the browser tree or listed in the viewer. d. Right-click the object and click **Audit Trail Events**.
Review changes to the UOM database	a. In the navigator, click **Units of Measure**. b. Click **File** > **Audit Trail Events**.

2. In the table displayed in the AF Audit Trail window, review changes made to PI AF data. The following columns are displayed:

Column	Description
Date	The date and time of the change.
Action	The type of change (Insert, Update, or Delete).
Type	The type of object that changed.
Database	The PI AF database containing the changed object.
Path	The hierarchical path to the changed object (when the object is a child of a parent object, the path shows the parent object).

Column	Description
Name	The name of the changed object.
User	The user who made the change, in the form of *domain\user*.

a. You can modify the content of the table, as follows:

To ...	Do this ...
Adjust the time period	a. Change **Start Time** and **End Time** to adjust the time period. You can choose from the following actions: ▪ Click to choose or enter a new date and/or time. ▪ Click to construct a relative time expression. b. Click (or press Enter) to display the updated table.
View changes for a subsequent time period	Click .
View changes for a previous time period	Click .
Adjust the number of change rows for current time period	a. Enter the number of rows you want to view in the **Maximum** field. b. Click (or press Enter) to display the updated table. **Note:** You can use **Maximum** to control response time. In addition, if the number of changes in the current time period exceeds **Maximum**, you can increase its value to display them.
Filter the returned results	a. Click to select a filter type. b. Enter text in the **Filter** field. c. If necessary, click . The table displays only those rows that contain your filter criteria.

3. Optional. To view details for a specific row, double-click it.

   a. In the AF Audit Trail Details window, review the details of the row you selected. In addition to **Path\Name**, **Indentity** and **Date**, the following information is displayed:

Column	Description
Action	Type of change (Insert, Update, or Delete) and the PI AF object sub-object (such as an attribute or attribute value).
Name	The PI AF sub-object; can be blank.
Id	The identity of the object referenced in the detail record. The value is either a GUID or an integer. The identity can be useful in situations where an object is renamed, since the underlying identity does not change.

Column	Description
Property Name	The column name from the SQL table in PI AF SQL Database (PIFD) for the changed sub-object. While this name is not necessarily easily understood, it provides a method for including all data within the record that has been changed.
Old Value	The value before the change.
New Value	The value after the change.

b. Click **Close** to return to the AF Audit Trail window.

4. Optional. You can export selected data grid rows or all the rows from the time range (if no rows are selected) in CSV format:
   a. In the **Export to File** field, enter a name for the export file.
   b. Click ... to open the Save As window and navigate to a specific directory.
   c. If you want to export the data from Audit Trail Details window for each selected row, select the **Detailed** check box.
   d. Click **Export All** or **Export Selected**, depending on whether you selected specific rows. (A tooltip indicates the number of rows selected.) The status bar initially indicates the total number of rows and then appends the time when the export completed. Note that while the export is executing, you can cancel it by clicking **Cancel Export**. The status bar indicates the time when the export was canceled.
5. To exit the Audit Trail utility, click **Close**.

View installed plug-ins

You check on installed plug-ins in the PI AF Server Properties window.

Procedure

1. Select **File** > **Server Properties**.
2. In the PI AF Server Properties window, click the **Plug-Ins** tab.
   - Move the pointer over a plug-in to see whether it is loaded and the version loaded.
   - Right-click a plug-in and select **Properties** to view details, such as the assembly, version, loaded version, support assemblies, and so on.

> **Note:**
> The loaded plug-in version usually matches the version on the PI AF server. However, if the version on the server changes, you must restart PI System Explorer to load the new plug-in version on the server.

Command-line plug-in registration

You can register different Plug-In implementations and support assemblies on the command line, using the **`RegPlugIn`** and **`RegPlugIn64`** utilities. For a list of parameters, see RegPlugIn parameters.

The Plug-In registration utilities are located in the `\PIPC\AF` folders for their corresponding `Program Files` directories:

```
\Program Files\PIPC\AF\RegPlugIn64.exe
\Program Files (x86)\PIPC\AF\RegPlugIn.exe
```

Examples

In the following examples, *MyPISystem* is the PI AF server where the Plug-In is to be registered, *MyPlugIn.dll* is the name of the Plug-In assembly, and *Support.dll* is the name of a support assembly for the Plug-In.

- .NET 3.5 Only Plug-In Targeting Any CPU

  The simplest to register, this implementation works on any operating system architecture and both versions of the .NET Framework. It cannot use new features that are specific to the .NET 4 version of AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll
  ```

- .NET 4 Only Plug-In Targeting Any CPU

  This implementation works on any operating system architecture and the .NET 4 Framework. It does not work with the .NET 3.5 version of the AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\4.0\MyPlugIn.dll
  ```

- .NET 3.5 and .NET 4 Plug-In Targeting Any CPU

  This Plug-In has two implementations. One targets the .NET 3.5 Framework and works with the .NET 3.5 version of AF SDK, and the other targets the .NET 4 Framework and works with the .NET 4 version of AF SDK. Both implementations are registered with the following command:

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\4.0\MyPlugIn.dll
  ```

- .NET 3.5 Only Plug-In Targeting 32- and 64-Bit Platforms

  This Plug-In has two implementations for the .NET 3.5 Framework. One implementation targets 32-bit operating systems and the other one targets 64-bit operating systems. With both implementations registered, the Plug-In works on any operating system architecture and both versions of the .NET Framework, but cannot use the new features that are specific to the .NET 4 version of AF SDK.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\x86\MyPlugIn.dll
  ```

  ```
  RegPlugIn64 /PISystem:MyPISystem PlugIns\x64\MyPlugIn.dll
  ```

- .NET 3.5 and .NET 4 Plug-In Targeting 32- and 64-Bit Platforms

  This Plug-In has four implementations. Two implementations are for the .NET 3.5 Framework, one for 32- and one for 64-bit platforms. The other two implementations are for the .NET 4, one for 32- and one for 64-bit platforms. With all implementations registered, it works on any operating system architecture and both versions of the .NET Framework.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\x86\MyPlugIn.dll PlugIns
  \4.0\x86\MyPlugIn.dll
  ```

  ```
  RegPlugIn64 /PISystem:MyPISystem PlugIns\x64\MyPlugIn.dll PlugIns
  \4.0\x64\MyPlugIn.dll
  ```

- .NET 3.5 Only Plug-In and Support Assembly Targeting Any CPU

  This implementation, with its support assembly, works on any operating system architecture and both versions of the .NET Framework. It cannot use the new features that are specific to the .NET 4 version of AF SDK.

```
RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\Support.dll
```

- .NET 3.5 and .NET 4 Plug-In and Support Assembly Targeting Any CPU

  This Plug-In has two implementations with two implementations of its support assembly. One targets the .NET 3.5 Framework and works with the .NET 3.5 version of AF SDK. The other targets the .NET 4 Framework and works with the .NET 4 version of AF SDK. Both implementations along with their support assemblies are registered with the following command.

  ```
  RegPlugIn /PISystem:MyPISystem PlugIns\MyPlugIn.dll PlugIns\Support.dll PlugIns
  \4.0\MyPlugIn.dll PlugIns\4.0\Support.dll
  ```

RegPlugIn parameters

The following table lists available parameters for the **RegPlugin** utility.

Parameter	Short form	Description
/PISystem:*string*	**/P**	Specifies the PI AF server to process. If not specified, the default PI AF server is used.
/OutputFile:*string*	**/O**	Specifies an output file. If specified, a SQL script is generated to create Plug-Ins. Use the **AppendFile** parameter to append to an existing file.
/AppendFile[-]	**/A**	Used in conjunction with the **OutputFile** parameter, specifies whether to append to an existing output file.
/List	**/L**	Lists registered assemblies in the specified PI AF server.
/Recursive[-]	**/R**	Automatically processes support assemblies in subdirectories based upon assembly and directory names. Use the **RootDirectory** parameter to specify a root directory that is different than the current working directory.
/Force[-]	**/F**	Forces registration of assemblies that are older than currently registered assemblies or forces the deregistration of assemblies that do not appear to be registered.
/Silent[-]	**/S**	Silent mode. Prevents messages from being displayed.
/Unregister	**/U**	Removes assembly and plug-ins from the specified PI AF server.
/RegFile:*string*	**/RF**	Creates a registration XML file for the specified assembly that can be used to register a plug-in assembly and all its related support assemblies. Any additional files specified are treated as support assemblies.
/Version	**/V**	Displays version information. All other parameters are ignored.
/User:*string*	**/user**	Specifies a different Windows user account to connect to the PI AF server.
/Password:*string*	**/PW**	Used in conjunction with the **User** parameter, specifies the password of a different Windows user account to connect to the PI AF server.

Parameter	Short form	Description
/Owner:*string*	**/Own**	Provides the owner file name of all specified support assemblies. Normally used with the **Directory** parameter when registering support assemblies. By default, the directory name will be determined relative to the **RootDirectory** parameter or current working directory if the root directory is not specified.
/Directory:*string*	**/Dir**	Provides the directory name for all specified support assemblies. Normally used with the **Owner** parameter when registering support assemblies. By default, the directory name is determined relative to the **RootDirectory** parameter or current working directory if the root directory is not specified.
/RootDirectory:*string*	**/RootDir**	Registers all assemblies relative to the specified root directory name. Normally used with the **Recursive** parameter when registering assemblies instead of using the **Directory**parameter. By default this parameter is set to the current working directory.
/Exclude:*string*	**/E**	Specifies input files or directories to exclude when searching a directory for files. You can specify this parameter more than once, but values must be unique.
files		Specifies the input assembly files, registration files, or directories to process. You can use the following wildcard specifiers to filter files processed in directories: • '*' matches zero or more characters • '?' matches exactly one character You can also use the **Exclude** parameter to filter the processed files.
@*file*		Provides additional input parameters to the specified file. The file should contain one parameter per line. Comment lines start with the '#' character.
/?, **/help**		Displays these option descriptions.

Create an XML registration file

For complex or frequently-executed registrations, you can create an XML file that contains the required settings.

Procedure

- To create the XML file, invoke `RegPlugin`, specifying both the required settings and the XML file name using the **`/RegFile:`** (**`/RF:`**) parameter.

Example

To create an XML file that registers a .NET 3.5 and .NET 4.0 version of `MyPlugIn.dll`, issue the following command:

```
RegPlugIn /RF: PlugIns\MyPlugIn.dll PlugIns\4.0\MyPlugIn.dll
```

The resulting registration file can be edited to supply any additional information required for the registration of the plug-in (not normally necessary). For 64-bit plug-ins, the registration file must be edited to set `LoadProperties` to `x64` and ensure that `Directory` is set to `x64`:

```
<SupportAssembly>
  <FilePath>x64\AFDRTest32Bit64Bit.dll</FilePath>
  <ID>1e00000c-3228-366a-3809-737433324269</ID>
  <Name>AFDRTest32Bit64Bit</Name>
  <Description>AFDRTest32Bit64Bit.dll Support Assembly</Description>
  <Directory>x64</Directory>
  <LoadProperties>x64</LoadProperties>
</SupportAssembly>
```

To register both implementations using the registration file, issue the following command:

```
RegPlugIn /PISystem:MyPISystem MyPlugIn.xml
```

Create an SQL registration script

Procedure

- To create an SQL registration script, invoke the `RegPlugIn` utility, specifying the required details and the `/OutputFile` or `/AppendFile` parameter. The resulting script can be loaded into the SQL database using SQL Server Management Studio or executed from a command line using the **`sqlcmd`** utility, enabling you to install plug-ins on machines where the PI AF SDK is not installed.

  Note:

  For plug-in developers: The SQL script needs to be generated every time you compile a new version of the plug-in.

Example

For example, to create a script that registers a NET 3.5-only Plug-In targeting any CPU, issue the following command:

```
RegPlugIn /O:MyPlugIn.sql PlugIns\MyPlugIn.dll
```

To generate an SQL registration script named `MyPlugIn.sql` that registers two implementations of MyPlugIn, a .NET 3.5-only Plug-In targeting x86 and x64, issue the following commands:

```
RegPlugIn /O:MyPlugIn.sql PlugIns\x86\MyPlugIn.dll
RegPlugIn64 /A:MyPlugIn.sql PlugIns\x64\MyPlugIn.dll
```

To generate an SQL registration script from a previously-created XML registration file, specify the XML file name on the command line. For example:

```
RegPlugIn /O:MyPlugIn.sql MyPlugIn.xml
```

Register plug-ins with generated SQL scripts

You register a plug-in from Microsoft SQL Server Management Studio or from the command line.

Procedure

1. Launch Microsoft SQL Server Management Studio.
2. Choose **File** > **Open** > **File**
3. Browse to the script and load it.
4. Execute the script.
5. To run the script from the command line, invoke the **`sqlcmd`** utility, specifying the **`-i`** *inputfile* option with the path and name of the SQL script as *inputfile* and the connection settings required to connect to the database server.

Plug-in provider management

By default, you can run PI AF plug-ins from any provider. For increased security, you can configure PI AF so that only plug-ins from trusted providers are allowed to be loaded and executed in the client application (PI AF server 2.5 and higher). Plug-in providers are encouraged to code-sign their plug-ins using Authenticode technology. You control how plug-in security is enforced by setting the verify level.

Verification level

You can display the verification level used when plug-ins are loaded by issuing the **`afdiag`** command with no parameters. The verification level is listed in the Configuration Settings section of the resulting output as `PluginVerifyLevel`.

`/PlugInVerifyLevel` parameter

You set the verify level with the **`afdiag`** command, and specify the **`/PlugInVerifyLevel`** parameter. Valid levels are:

- `None`: Disable validation; run all plug-ins.
- `AllowUnsigned`: Run unsigned plug-ins and plug-ins with valid signatures.
- `RequireSigned`: Run only plug-ins with valid signatures.
- `RequireSignedTrustedProvider`: Run only plug-ins with a valid signature from a trusted provider.

Other **`AFDiag`** parameters for managing plug-in providers

The following **`afdiag`** parameters are available to manage plug-in providers:

- Display a list of trusted providers: **`/TrustedProviderList`**
- Add a trusted provider: **`/TrustedProviderAdd:`***providername*
- Remove a trusted provider: **`/TrustedProviderRemove:`***providername*

For more information on these parameters, see AFDiag utility parameters. You can locate the provider name by viewing the digital signature for the plug-in DLL, as described in Locate the names of trusted providers.

Locate the names of trusted providers

Determine the name of a trusted provider of a plug-in DLL by viewing the details of its digital signature. The string for the trusted provider name must be contained in the certificate's subject.

Procedure

1. Right-click the plug-in DLL and click **Properties**.
2. In the Properties window, click the **Digital Signatures** tab. (If the plug-in is unsigned, this tab is absent.)
3. Select a **Name of signer** in the **Signature list** and click **Details**.
4. In the Digital Signature Details window, click **View Certificate**.
5. In the Certificate window, click the **Details** tab and scroll to the **Subject** field.

   The value displayed for this field is the name of the trusted provider. You can you use any subset or the whole name of any of the names that are part of the subject as the *<providername>* variable for the **`/TrustedProviderAdd`** and **`/TrustedProviderRemove`** parameters.

Technical support and other resources

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or through the OSIsoft Tech Support Contact Us page (https://techsupport.osisoft.com/Contact-Us/). The website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Details about your computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

To ask questions of others who use OSIsoft software, join the OSIsoft user community, PI Square (https://pisquare.osisoft.com). Members of the community can request advice and share ideas about the PI System. The PI Developers Club space within PI Square offers resources to help you with the programming and integration of OSIsoft products.