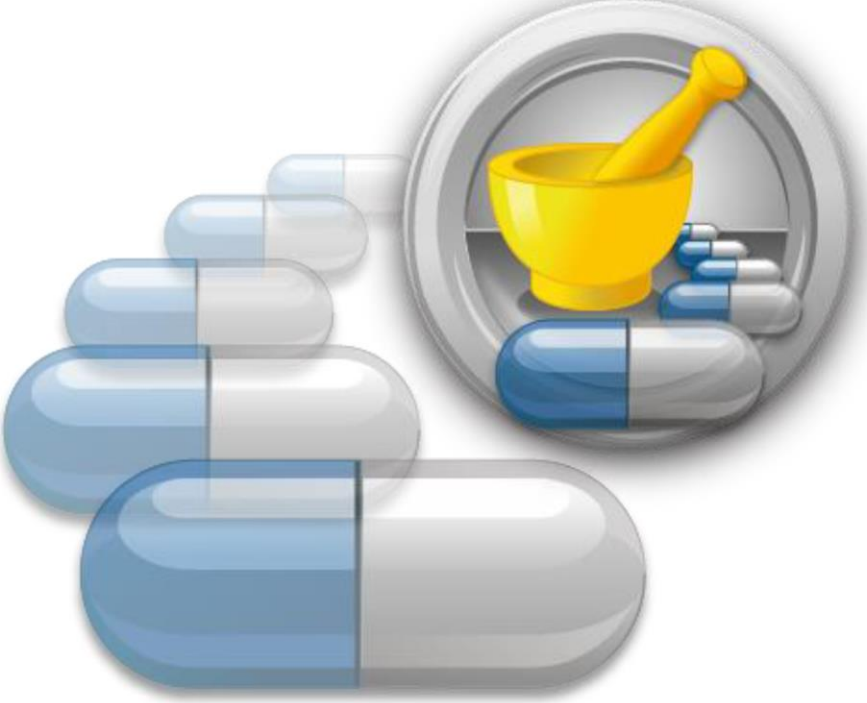
FT PharmaSuite®



# ADMINISTRATION

RELEASE 10.01.00

TECHNICAL MANUAL

PUBLICATION PSAD-RM010B-EN-E-MARCH-2021
Supersedes publication PSAD-RM010A-EN-E

Rockwell Automation

**Contact Rockwell** See contact information provided in your maintenance contract.

**Copyright Notice**

**Trademark Notices**

**Other Trademarks**

**Warranty** This product is warranted in accordance with the product license. The product's performance may be affected by system configuration, the application being performed, operator control, maintenance, and other related factors. Rockwell Automation is not responsible for these intervening factors. The instructions in this document do not cover all the details or variations in the equipment, procedure, or process described, nor do they provide directions for meeting every possible contingency during installation, operation, or maintenance. This product's implementation may vary among users.

This document is current as of the time of release of the product; however, the accompanying software may have changed since the release. Rockwell Automation, Inc. reserves the right to change any information contained in this document or the software at any time without prior notice. It is your responsibility to obtain the most current information available from Rockwell when installing or using this product.

# Contents

# Figures

# Introduction

This manual is intended for system administrators of PharmaSuite at a customer's site. The following chapters describe the methods of integrating custom hardware, best practices for managing user accounts of a client operating system, and monitoring features. You will learn how to connect custom scales, printers, and scanners to the system, export and import master recipes, master workflows, and building blocks, import equipment classes and entities from an external source, configure the device identifier, localize each as unit of measure, and which event sheets are used for various purposes.

The PharmaSuite client for the administration of database objects, such as access privileges, lists, applications, users, or user groups is PS Administration. For methods of controlling access to the system and how to define access rights, create users, user groups, and assign access rights to them, see "Implementation Guide PS Administration" [A1] (page 97).

The system administrators need to be familiar with the FactoryTalk® ProductionCentre functions for controlling access to the system, maintaining list objects, and using event sheets.

## Typographical Conventions

This documentation uses typographical conventions to enhance the readability of the information it presents. The following kinds of formatting indicate specific information:

**Bold typeface**

Designates user interface texts, such as

- window and dialog titles
- menu functions
- panel, tab, and button names
- box labels
- object properties and their values (e.g. status).

*Italic typeface*

Designates technical background information, such as

- path, folder, and file names
- methods
- classes.

CAPITALS

Designate keyboard-related information, such as

- key names
- keyboard shortcuts.

`Monospaced typeface`

Designates code examples.

# Extension and Naming Conventions

This section describes how to control efforts for migrating your PharmaSuite installation to upcoming versions. If you cannot observe the guidelines for technical reasons, please report the issue to your dedicated delivery team of Rockwell Automation or your system integrator.

PharmaSuite artifacts fall into two main groups: building blocks and PharmaSuite core artifacts.

- When you wish to modify a **building block**, use a copy of the building block. For details, see "Technical Manual Developing System Building Blocks" [A2] (page 97).
- When you wish to modify a **PharmaSuite core artifact**, the extension strategy depends on the artifact itself.
  Modify **DSX objects** and copy **all other PharmaSuite artifacts** (e.g. XML configurations for services).

In all cases, please follow the guidelines to control the migration effort (page 3).

## Guidelines to Control the Migration Effort

Please ensure that you observe the guidelines listed below.

1. Retain the published API
   The published API of PharmaSuite is accessible via the PharmaSuite start page ("PharmaSuite-related Java Documentation" [C1] (page 98)).
   - If you adapt PharmaSuite on Java level, you must only use the published PharmaSuite Java API.
     Published interfaces will only be changed if necessary. Changes to these interfaces and classes will be announced in future release notes.
   - Avoid using methods and classes that are not published, classes from the implementation package (*...impl...*), or Pnuts functions from any PharmaSuite subroutine.
     These classes and functions may change in future versions of PharmaSuite without notification.

2. Rather modify than copy
   Whenever you wish to extend/modify any object of PharmaSuite in Process Designer (DSX objects), just override the object, except for the following objects:

   - For the **Application** object **Default Configuration**, apply the mechanism of nested configurations in order to reduce a potential migration effort.
     PharmaSuite provides the PS Administration application to administer database objects such as access privileges, lists, applications, users, or user groups, see "Implementation Guide PS Administration" [A1] (page 97).
     Using Process Designer for these tasks is possible, but does not provide the full functional scope required for PharmaSuite. For information on the administration of database objects with Process Designer in this case, please refer to chapter "Managing Configurations" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97)).

   - For **FSMs** (flexible state models), structural changes (i.e. changes to states and transitions) are not allowed in order to enable a later system migration.
     Modifications of semantic properties can be applied to the standard FSM. They do not impact a system migration.

   When you migrate your PharmaSuite installation to another version, PharmaSuite Update and Migration displays a warning if a standard object was changed. Subsequently, you can decide whether to adapt your extension, ignore the changes delivered with the new version, or replace your extension with the new version (if applicable).

3. Mark your objects
   When naming your artifacts (e.g. objects in Process Designer, classes, interfaces, methods, functions, building blocks), use specific prefixes for your objects. The main purpose of the naming conventions is to prevent naming conflicts with deliverables from other vendors or with other versions.

   - Define and make use of a vendor code consisting of up to three uppercase letters as prefix (e.g. MYC for the My Company vendor code).
     The **X_** and **RS_** prefixes are reserved for PharmaSuite and PharmaSuite-specific product building blocks, respectively.

   - Do not reuse any of the prefixes of PharmaSuite objects in Process Designer in order to avoid conflicts during migration.

   - This guideline also applies to UDA definitions and column names of application tables.

   - Additional conventions apply to building blocks (page 5).

If you do not observe the guidelines, an update process during system migration may fail due to conflicts.

## Building Block-specific Conventions

Besides the general conventions (page 3), additional conventions apply to building blocks related to vendor code, version number, and length restrictions:

1. Vendor code
   - It must be appended to the name of a phase or parameter class used in the UI (enclosed in round brackets).
   - It must be used as prefix for the AT definitions.
   - The package name must also contain a vendor reference. You can either use the vendor code or write out the company's full name.
2. Version number
   - The version number consists of two components, an integral part to refer to a major version and a fractional part to refer to a minor version, e.g. 2.1.
   - It must be appended to the name of the phase or parameter class used in the UI and to the base name used for the generated artifacts.
   - For the UI, the version number is enclosed in square brackets, e.g. [2.1].
   - Internal names must not contain brackets and dots, since Java does not allow the usage of these characters. Therefore, the last four characters are reserved for the version number, with digits 1 and 2 representing the major version and digits 3 and 4 representing the minor version. The format is xxyy, e.g. 0201 for version [2.1], 0100 for version [1.0], or 0113 for version [1.13].
3. Length restrictions
   - The maximum length of the **name** of a phase or parameter class used in the UI is 64 characters.
   - The maximum length of the **base name** of a phase building block or parameter class is 18 characters (14 for the name, 4 for the version).

Examples:

- **Hello World** phase of **My Company** with vendor code **MYC** in version **2.1**

```
<Name>Hello World Phase (MYC) [2.1]</Name>
<PhaseLibBaseName>HelloWorld0201</PhaseLibBaseName>
<ATDefinitionPrefix>MYC</ATDefinitionPrefix>
<PackageName>com.mycompany.phase.helloworld</PackageName>
```

- **My Parameter** parameter class of **My Company** with vendor code **MYC** in version **1.0**

```
<Name>My Parameter (MYC) [1.0]</Name>
<ParamClassBaseName>MyParam0100</ParamClassBaseName>
<ATDefinitionPrefix>MYC</ATDefinitionPrefix>
<PackageName>com.mycompany.parameter.myparam</PackageName>
```

## Oracle Database Data Types: varchar2 and nvarchar2

When you define text fields for FactoryTalk ProductionCentre application tables or UDAs, you should be aware that there are significant differences between the **varchar2** and **nvarchar2** data types used by Oracle databases.

- The maximum field length for both database data types is restricted to 4000 bytes.
- A field of the **nvarchar2** data type works as expected: For a 4000 characters text field, you can only insert the maximum of 2000 2-byte UTF8 characters or 1300 3-byte UTF8 characters. PharmaSuite text input fields check the number of bytes.
- For an Oracle 11 database, a field of the **varchar2** data type can only handle 1000 2-byte UTF8 characters or 667 3-byte UTF8 characters. In this case, PharmaSuite text input fields check the byte length and prevent the database exception "ORA-01704: string literal too long".

TIP

Previous versions of FactoryTalk ProductionCentre (prior to 9.3) and PharmaSuite (prior to 5.0) contained **varchar2** database data type definitions instead of **nvarchar2**. Therefore a migrated PharmaSuite system may contain **varchar2** definitions and the maximum length of a **varchar2** database field may be defined in maximum number of **bytes**. In this case, if a text does not only contain 1-byte UTF8 characters, the **byte** length is greater than the **char** length and can cause a "value too large exception" (ORA-01401: inserted value too large for column, ORA-12899: value too large for column).
For this reason, we recommend to migrate all **varchar2** fields to **nvarchar2** fields. A migrated FactoryTalk ProductionCentre database should have the same database schema as the database of a newly installed FactoryTalk ProductionCentre system and only contain **nvarchar2** database data type field definitions.

Chapter 3

# Integrating Custom Hardware

PharmaSuite provides interfacing to several qualified scales, printers, and barcode scanners. In case it is required to interface with equipment that has not been certified for use with PharmaSuite, integration and testing of those pieces of equipment can be done by qualified system integrators.

Regarding printers, we expect all standard Windows printer drivers to function correctly with PharmaSuite. We recommend, however, to verify and test printers and drivers before using them in a production environment.

Barcode scanners must be able to emulate keyboard input to be suitable for use with PharmaSuite.

## Integrating New Scales

Before you can use a scale with PharmaSuite you need to configure the scale itself:

- Mettler Toledo SICS protocol-based scale (page 9)
  - AB, AG, PB, PL, PG, PR, SB, SG, SR (with built-in RS232 (V.24) interface)
  - XP (with non-built-in RS232C (V.24) interface)
  - ICS4_5, XP, XS (with built-in Ethernet 10Base-T (RJ45) interface)
- Mettler Toledo SICS protocol-based terminal (page 11)
  - IND780 (with built-in RS232 (V.24) serial port)
  - IND890 (with built-in Ethernet 10Base-T (RJ45) interface)
- Mettler Toledo scale terminal
  - ID1 [Plus/Minus], ID2, ID3[s] (MMR protocol-based) (page 15)
  - ID5 and ID7 (MMR protocol-based) (page 16)
  - IND (MMR protocol-based) (page 17)
- Mettler Toledo scale interface
  - GD12 (IDNet protocol-based) (page 20)
- Sartorius xBPI protocol-based scale (page 20)
- Sartorius SBI protocol-based scale (page 22)

and to define and configure the scale equipment entity in PharmaSuite (page 25).

As a first step, identify the exact make and type of the scale, with which type of interface it connects to a PC, and which communication protocol it uses. Typically, you will find this information in the scale manual or on the manufacturer's website. PharmaSuite provides drivers for Mettler Toledo [D1] (page 98) and Sartorius [D2] (page 98) scales that

- can be connected to a PC by serial RS232 (V.24) or (if available) Ethernet network (TCP/IP) interfaces and
- use one of the following communication protocols:
  - SICS, MMR, or IDNet (Mettler Toledo)
  - SBI/xBPI (Sartorius)
    and are included in the range of supported scale families.

Support for some features, such as identification by serial number, calibration, blocking of scales keyboard, and transfer of nominal value depends on

- the scale driver
- the scale type
- the setup of the scale
- the configuration of the scale equipment entity
- how the scale is used by the Dispense application.

**TIP**

If you need to operate scales for which there is no scale driver provided by PharmaSuite, please refer to the PharmaSuite documentation. In particular, see chapter "Adding New Scale Drivers" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97) and the documentation on the PharmaSuite SDK [A9] (page 97).

## Configuring Scales

To configure the new scale, proceed as described in the section that applies to your scale type.

### METTLER TOLEDO SICS PROTOCOL-BASED SCALES

To connect Mettler Toledo scales that use the SICS communication protocol, proceed as follows:

- Configuration of scales with a built-in RS232 (V.24) interface (METTLER AB, AG, PB, PL, PG, PR, SB, SG, SR):

  1. Change the interface mode from **Printer** to **Host**.
  2. Set the parameters as follows:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | Even parity |
| Communication | 2400 baud |
| Hardware handshake | None |

- Configuration of scales with non-built-in RS232C (V.24) interface (METTLER XP).
  Set up the interface mode of the scales as follows:

| Parameter | Value |
|---|---|
| Host | RS232 fix |
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | Even parity |
| Communication | 2400 baud |
| End of line | <CR> <LF> |
| Charset | Ansi/Win |
| Continuous mode | off |
| Output format | MT-SICS setting |
| Handshake | None |

- If only one scale is connected per COM interface, type **0** (zero) in the **Terminal number** property of the scale configuration data of the scale equipment entity in Data Manager - Equipment.

TIP - NON-DEFAULT COMMUNICATIONS SETTINGS

See "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

TIP - SICS LEVEL

See "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

TIP - MULTIPLE SCALES

Mettler industrial terminals typically allow to connect to more than one scale. If the terminal supports the MMR protocol, use the IND driver. Otherwise, if the terminal supports the SICS protocol, use the SICS driver. Be aware that some terminals have no standard SICS command to switch between multiple scales connected to the same terminal. This does not apply to the IND780 or IND890 terminal, see section "Mettler Toledo SICS Protocol-based Terminals" (page 11).

- Configuration of scales with a built-in Ethernet 10Base-T (RJ45) interface (e.g. METTLER ICS4_5, XP, XS with Ethernet card/option)
  Set the parameters of the scales as follows:
  (**Communication - COM2** (use the COM# assigned to Ethernet not to RS232C))

| Parameter | Value |
|---|---|
| Mode | Dialog |
| Parameter Ethernet | Contact your network administrator for the following values:<br>DHCP, Local IP, Subnet mask, Gateway |
| TCP mode | TCP mode: Server, Local port: 4305, Disconnect timeout: 0 |

Please make a note of the values of **Local IP** and **Local port**, you will need them during the scale configuration in Data Manager - Equipment.

### Mettler Toledo SICS Protocol-based Terminals

To connect Mettler Toledo terminals (e.g. IND780 or IND890, see also [D3] (page 98), [D7] (page 98)) that use the SICS host communication protocol, proceed as follows:

- Configuration of IND780 terminals with a built-in RS232 (V.24) serial port COM1 or COM2 in the Terminal Setup Mode:
  1. Select **Communication\Connections/X4** and insert a connection for the respective port (e.g. **COM1** or **COM2**) with **Assignment=SICS**, and **Trigger = None**, and **Template = None**.
  2. Select **Communication\Interfaces\Serial/X4** and modify the communication parameters for the respective serial port (COM1 or COM2) as follows:

| Parameter | Value |
|---|---|
| Data bits | 7 data bits |
| Interface/Hardware | RS-232 |
| Parity | Even parity |
| Baudrate | 2400 baud |
| Flow control/Handshake | None |
| Character set | 1252 |
| String/Frame | <CR><LF> |

- Configuration of IND890 terminals with a built-in Ethernet 10Base-T (RJ45) interface
  Set the parameters of the terminal as follows:
  (**Communication\Connections – insert a connection for Port ENet**)

| Parameter | Value |
|---|---|
| Port | Enet |
| Hardware | Ethernet |
| Assignment | SICS |
| Trigger | None |
| Timeout | 0 seconds |
| Template | None |

Set the parameters of **Communication\Network\Ethernet\Settings** as follows:

| Parameter | Value |
|---|---|
| DHCP, IP address, Subnet mask, Gateway address | Contact your network administrator for the values. |
| Primary port | 1701 |

Please make a note of the values of **IP address** and **Primary port**, you will need them during the scale configuration in Data Manager - Equipment.

- Terminals allow to connect to more than one weighing platform. The scale identifier number (1..5) must be provided as the **Terminal number** property of the scale configuration data of the scale equipment entity in Data Manager - Equipment. If only one scale is connected per COM interface, type **0** (zero) for the property.

TIP - NON-DEFAULT COMMUNICATIONS SETTINGS

See "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

TIP - SICS LEVEL

See "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

TIP - MULTIPLE SCALES AT THE SAME TERMINAL

**SNS n** is the SICS command to switch between multiple scales connected to the same terminal. However, this command is currently supported only by IND780 with the latest firmware level. Alternatively, **AW 010 n** is the equivalent SICS command supported by IND890 with a firmware level v1.30 or higher. If your terminal does not support the commands and you have more than one scale connected to such a terminal, we recommend to set up the MMR protocol and use the IND driver instead. If this is also not supported, a customer specific driver is needed.

TIP - ETHERNET COMMUNICATION

Even though IND780 has a built-in Ethernet port it cannot be configured for connections using the SICS protocol.

However, IND890 supports running the SICS protocol over Ethernet.

TIP - DISABLING KEYBOARD OF TERMINAL VS. SCALE

IND780 and IND890 allow to disable the keyboard of the terminal through the driver. PharmaSuite will automatically disable the keyboard while it uses a scale, if the **Disable scale keyboard** and **Scale keyboard lockable** properties are set in the scale configuration data of the scale equipment entity in Data Manager - Equipment.

We recommend this setting to prevent changes to the weighed value that are not controlled by PharmaSuite e.g. zeroing or taring directly at the terminal. If, however, scales connected to the terminal have a keyboard of their own (e.g. Laboratory scales connected via SICS on COM2) this keyboard will not be disabled. We recommend to either disable this keyboard at the scale itself, block it mechanically, or train the operators to not use it.

**TIP - ENABLING DELTATRAC/SMARTTRAC GRAPHICAL DISPLAY**

The terminal supports a weighing aid in the form of a bar graph or +/- display to visualize the deviation of the actual weight against the target value and its tolerance values. PharmaSuite can control this automatically after taring if
1) the **Send nominal value** property is set in the scale configuration data of the scale equipment entity in Data Manager - Equipment and
2) this feature has been set up in IND780 (SmartTrac size other than **none** must be selected at **Setup** > **Terminal** > **Display**, SmartTrac graphic display type must be selected for the scale at **Setup** > **Application** > **Operation** > **Target** > **Scale #**). IND890 has the DeltaTrac feature always enabled.

**TIP - SCALE CALIBRATION AND ADJUSTMENT**

This SICS level 3 functionality is not supported by IND780 or IND890.

### Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals

The following tips apply to Mettler Toledo SICS protocol-based scales and terminals:

**TIP - SERIAL NUMBER OF THE TERMINAL VS. ACTIVE SCALE**

Terminals and scales support to query for their serial numbers. After the scale has been selected successfully, PharmaSuite checks if the serial number of the connected scale or terminal corresponds to the serial number defined in the basic data of the scale equipment entity in Data Manager - Equipment. If the serial number of the connected scale is not available (e.g. for an old IDNet weighing bridge), the serial number of the terminal is used instead for the check.

The serial number is not evaluated by any of the PharmaSuite product phases.

**TIP - NON-DEFAULT COMMUNICATIONS SETTINGS**

If the default communications settings listed above cannot be used, configure the driver to use non-default communications settings. For this purpose, in Process Designer, open the **ScaleDrivers List** object and locate the **SICS** item. **SICS** uses the default settings as listed above. When you add other items with a "qualifier" indicated by a slash (**/**) after the name (e.g. **SICS/2400-7-EVEN-1-NONE**), the same driver will be loaded, but it will use the qualifier indicated after the slash in the name (e.g. **2400-7-EVEN-1-NONE**) to set the communications settings.

The qualifier can consist of several options in the following order: **baud rate**, **number of data bits**, **parity**, **stop**, and **flow**. The values of the **parity** options are **NO**, **EVEN**, and **ODD**, the values of the **flow** option are **NONE**, **HARDWARE**, and **XONXOFF**. The options are separated by a dash (**-**) or an underscore (**_**).

Examples: **SICS/2400** runs with 2400 baud and the default settings of the other options. With **SICS/2400-7-EVEN** you specify baud, data, and parity and use the default settings of the other options.

**SICS/9600-8-No-1-NONE** is a pre-configured name of the SICS driver intended for SICS-based scales or terminals using the factory default setting of the device.

We recommend to specify each option. Make sure to restart PharmaSuite for Production Execution after having saved the changed list in order to make the changes effective.

**TIP - SICS LEVEL**

The PharmaSuite scale driver expects that the scale has implemented at least SICS level 0 and 1. A scale that has implemented only a subset of the SICS level-1 commands (e.g. TAC command is not supported), does not provide the complete functionality expected by the scale driver.

To retrieve the SICS levels with their versions and the SICS commands implemented by your scale, activate the INFO log level to log the replies to the I0, I1, I2, and I3 commands (see section "Troubleshooting Scale Communication" (page 26)).

### Mettler Toledo Scale Terminals ID1 [Plus/Minus], ID2, and ID3[s] (MMR Protocol-based)

This section applies to the ID1 [Plus/Minus], ID2, and ID3[s] Mettler Toledo Industrial Scale Terminals, which use the MMR communication protocol.

To configure scale terminals with a built-in RS232C (V.24) interface, set the parameters as follows:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | Even parity |
| Communication | 2400 baud |
| Hardware handshake | NONE |
| Mode | Dialog |

For details how to configure the driver to use non-default communication settings, see TIP - NON-DEFAULT COMMUNICATIONS SETTINGS in section "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

The ID1 multirange scale terminal provides a CL 20mA output via Interface 101 and an RS232C (V.24) interface via Interface 102. These interfaces are used to connect to a computer. ID3s corresponds to ID1 with numeric keyboard. Interface 301 is used for 20mA and Interface 302 for RS232C (V.24). Only one weighing header can be connected to an ID1 terminal. Unlike ID1, ID3 has a numeric keyboard.

DeltaTrac can be controlled if plus/minus weighing is configured for the terminal and a nominal value and range are transmitted to the scale.

To populate the bar diagram of the terminal with the current nominal value after taring, set Optical weighing-in aid for ID1/ID3 in the master mode.

| Parameter | Value |
|---|---|
| PlusMinus weighing | YES |
| Application | Weighing-in to target weight |
| Display | Normal or difference between set weight and actual weight |
| LED | LED-analog display as lighting trunking |

**TIP**

The serial number and calibration features are not supported.

### Mettler Toledo Scale Terminals ID5 and ID7 (MMR Protocol-based)

This section applies to the ID5 and ID7 Mettler Toledo Industrial Scale Terminals, which use the MMR communication protocol.

To configure scale terminals with a built-in RS232C (V.24) interface, set the parameters as follows:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | Even parity |
| Communication | 2400 baud |
| Hardware handshake | NONE |
| Mode | Dialog |

For details how to configure the driver to use non-default communication settings, see TIP - NON-DEFAULT COMMUNICATIONS SETTINGS in section "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

For ID5 terminals, the settings listed above correspond to the factory preset state. The interface parameters preset for the ID7 terminal, however, differ from them and have to be adjusted accordingly.

An ID5 multirange scale terminal must be provided with an RS232 (V.24) or 20 mA current loop interface by means of installing the options 082, 084, or 089 in module slots 1 or 2, which connect to the computer. By using a GD31 multiplexer you can connect up to three weighing headers to an ID5 terminal. For this purpose, the individual scales are arranged with multiple ID5 scale equipment instances at the same work center and the same COM interface, but have different terminal numbers "1", "2", or "3" maintained in the scale configuration data of the scale equipment entity in Data Manager - Equipment.

This terminal class can connect to more than one weighing bridge per terminal. Each scale requires its individual scale equipment entity in Data Manager - Equipment using the same COM port but different terminal numbers.
DeltaTrac can be controlled if plus/minus weighing is configured for the terminal and a nominal value and range are transmitted to the scale.

**TIP**

The serial number and calibration features are not supported. A dosage control module DOSPac or other modules such as FORMPac that can optionally be built into an ID5 terminal are not supported by the driver.

### METTLER TOLEDO SCALE TERMINAL IND (MMR PROTOCOL-BASED)

This section applies to the IND Mettler Toledo Industrial Scale Terminals, which use the MMR communication protocol.

- To configure a scale terminal with a built-in RS232C (V.24) interface, set the parameters as follows:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | Even parity |
| Communication | 2400 baud |
| Mode | Dialog |
| Dialog Mode | MMR |
| Hardware handshake | No handshake |
| AUTOMATIC CONTINUOUS TRANSMISSION | NO AUTO TRANSMISSION |
| STRING FRAMING | CR Yes, LF Yes, <STX>---<ETX> No, BLOCK CHECK CHAR No |

The settings listed above do not correspond to the factory preset state (8 bits / 1 stop bit/ no parity, 9600 Baud). Thus, the settings need to be adjusted accordingly. For details how to configure the driver to use non-default communication settings, see TIP - NON-DEFAULT COMMUNICATIONS SETTINGS in section "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14). For an IND terminal, it is also necessary to setup the MMR communication protocol for use with the host communication.

#### EXAMPLE CONFIGURATION OF IND690 TERMINALS

To configure an IND690 terminal with a built-in RS232C (V.24) interface, set the parameters as follows (assumption: connection port of the IND690 terminal to the PC is COM1):

| Parameter | Value |
|---|---|
| Mode->Interfaces-> COM1->RS232-> Communication | --- |
| BITS/CHAR | 7 |
| STOP BITS | 1 |
| PARITY | Even |

| Parameter | Value |
|---|---|
| BAUDRATE | 2400 |
| Hardware handshake | NONE |
| Mode->Interfaces-> COM1->RS232->MODE-> DIALOG MODE->MMR | --- |
| HANDSHAKE | NO HANDSHAKE |

To connect scales to the terminal, e.g. an IDNET-based scale (= scale_1) with an integrated scale processor on COM2 and a SICS-based scale (=scale_2) on COM3, set the parameters on the IND690 terminal as follows (assumption: COM2 of the terminal has an IDNET interface, COM3 of the terminal has a SICS interface):

| Parameter (connect scale_1 to IND690) | Value |
|---|---|
| Mode->Scale->(CH:1) | --- |
| all parameters | keep standard values |

| Parameter (connect scale_2 to IND690) | Value |
|---|---|
| Mode->Scale->(CH:2) | --- |
| all parameters | keep standard values |
| Mode->Interfaces-> COM3->RS232-> SICS-SCALE-> Communication | --- |
| BITS/CHAR | 7 |
| STOP BITS | 1 |
| PARITY | EVEN |
| BAUDRATE | 2400 |
| Hardware handshake | NONE |

The IND scales terminal has an RS232C (V.24) or 20 mA current loop interface (typically labeled COM1/RS232C built in), which connects to the computer. By using the appropriate optional interface modules, you can connect up to four weighing headers to an IND terminal. For this purpose, the individual scales are arranged with multiple IND scale equipment instances at the same work center and the same COM interface, but with different terminal numbers "1", "2", or "3" maintained in the scale configuration data of the scale equipment entity in Data Manager - Equipment. For further information on scale equipment entities, please refer to the "Data Manager User Documentation" [C2] (page 98).

This terminal class can connect to more than one weighing bridge per terminal. Each scale requires its individual scale equipment entity in Data Manager - Equipment using the same COM port but different terminal numbers.
DeltaTrac can be controlled if plus/minus weighing is configured for the terminal and a nominal value and range is transmitted to the scale.

- To configure an IND scale terminal with a built-in Ethernet 10Base-T (RJ45) interface (e.g. Ethernet 960 card), set the parameters as follows:
  (**Master mode - Interfaces - COM4** (use the COM# assigned to Ethernet))

| Parameter | Value |
|---|---|
| Ethernet<br>Mode | Dialog<br>1 Dialog Mode - MMR<br>Dialog Mode - MMR- Handshake - No Handshake<br>Dialog Mode - MMR- Autorepeat - No Autorepeat<br>Dialog Mode - MMR- Transfer String - Standard Setting |
| Ethernet<br>Communication | Standard settings |
| Ethernet<br>IP, Subnet & Gateway address | Contact your network administrator for the following values:<br>IP, Subnet mask, Gateway |

To configure the TCP parameters, open a browser and navigate to the configured IP address. Then open the **Setup** menu (SETUP Port 0 (High-Speed Serial) - TCP/IP Mode Local Port) and set the port number (default: 8000).

Please make a note of the value of **IP**, you will need it during the scale configuration in Data Manager - Equipment.

**TIP**

The serial number feature is supported. The number is taken over from the terminal, not from the scale attached to the terminal.

The calibration feature is not supported.

### Mettler Toledo Scale Interface GD12 (IDNet Protocol-based)

To connect Mettler Toledo scale interfaces that use the IDNet communication protocol, you need to be aware that configuration of the adapter is not possible since the parameters of the built-in RS232C (V.24) interface are fixed with the following settings:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 2 stop bits |
| Communication | Even parity |
| Communication | 9600 baud |
| Hardware handshake | NONE |

This adapter connects to one weighing bridge.

TIP

The serial number and calibration features are not supported.

### Sartorius xBPI Protocol-based Scales

To connect Sartorius scales that use the xBPI communication protocol, proceed as follows:

- Configuring the RS232 (V.24) interface:
  - Select xBPI as interface protocol, e.g. for the ISI-10 or Combics terminal.
  - Based on the factory settings, the following settings must be configured in the setup menu (If all segments light up during the closing operation, press the ENTER key):
    Type the factory default password **202122** and set **S-3.1 COM XBPI-232** as the system value when the computer is connected to the serial interface 3 of the ISI-10 terminal (S3).
    In this case, the following configuration is strictly defined for the xBPI protocol and a setting is not necessary or possible, neither at the scale nor in Data Manager - Equipment. If only one scale is connected to the terminal, type **0** (zero) in the **Terminal number** property of the scale configuration data of the scale equipment entity in Data Manager - Equipment. Otherwise, type the number (1..3) of the respective scale connected to the terminal.

- Set up the interface mode of the scales as follows:

| Parameter | Value |
|---|---|
| Communication | 8 data bits |
| Communication | 1 stop bit |
| Communication | odd parity |
| Communication | 9600 baud |
| Hardware handshake | no |

- Configuring the RS485 interface:
  Sartorius scales using the xBPI protocol (e.g. IS platforms) will be connected to weighing work station PCs with RS485 interface via a point-to-point configuration (half duplex) to bypass greater distances (20..1000 m).
  Change the **RS232C** interface (e.g. YD052IS) of the scale to **RS485**, e.g. with YD051IS (*Sartorius data interface RS485 with 12-pin round connector, order no. 69IS 009*).
  Networking several weighing platforms with an RS485 bus system/party-line configurations with more than one scale connected to the RS485 bus are not supported.
- Configuring the Ethernet 10Base-T (RJ45) interface:
  - The optional UniCOM interface is required.
  - Configure the parameters in the Combics **Setup** menu as follows:

| Parameter | Value |
|---|---|
| Data Protocol Ethernet | Contact your network administrator for the following values: Source IP, Source name, List port, Subnet mask, Gateway |
| Data Protocol Ethernet Protocol | TCP |
| Data Protocol Ethernet Mode | XBPI |

Please make a note of the values of **Source IP** and **List port**, you will need them during the scale configuration in Data Manager - Equipment.

- Some terminals allow to connect to more than one weighing platform. The scale identifier number (1..3) must be provided as **Terminal number** property of the scale configuration data of the scale equipment entity in Data Manager - Equipment. If there is only one scale connected per COM interface, type **0** (zero) for the property.

### Sartorius SBI Protocol-based Scales

These scales are provided with an RS232 (V.24) interface. They must be configured with the corresponding setup settings for continuous display without stoppage and long data format.
Only scales that have a **zeroing key** (**T** key) as well as an **F** key, which can be assigned to the tare memory function, can be tared through the PharmaSuite - Dispense application.
Only those scales with a numeric keyboard can be supplied with manual tare values by the Dispense application (called **applicative tare 2** by Sartorius). This requires knowing whether and which *data input option* and *keyboard overlay* are installed. Otherwise, the PharmaSuite - Dispense application is unable to tare the scale and the scale cannot be used.

- Configuration of the RS232 (V.24) interface:

| Parameter | Value |
|---|---|
| Communication | 7 data bits |
| Communication | 1 stop bit |
| Communication | even parity |
| Communication | 2400 baud |
| Hardware handshake | no |

For details how to configure the driver to use non-default communication settings, see TIP - NON-DEFAULT COMMUNICATIONS SETTINGS in section "Tips Specific to Mettler Toledo SICS Protocol-based Scales and Terminals" (page 14).

### Example: Scale menu settings for QC 7 CCE-S

The settings are similar for other families.
Default settings are marked by an asterisk (*).
Use the **SBI4QC** scale driver.

```
9 - - 1 Factory defaults
5 1 5 2400 Baud
* 7 Bit
5 2 4 Even parity
*5 3 1 1 stop bit
*5 4 2 Hardware handshake with 2 characters after CTS
6 1 4 Autoprint regardless of stability
6 2 1 Print stoppable
*7 2 2 With ID code for data output (22 character format)
7 1 2 Automatic output of the application parameters
7 3 2 Automatic output of tare memory data
2 2 2 Program tare memory
```

### Example: Scale menu settings for Sartorius Masterpro series in SBI mode

Typically, the scales of the Masterpro series (e.g. Sartorius LA 2200) will be used in XBPI and not in SBI mode and use the XBPI driver instead.
The settings are similar for other families.
Default settings are marked by an asterisk (*).
Use the **SBI4LA** scale driver.

```
9 - - 1 Factory defaults
5 1 5 2400 Baud
* 7 Bit
5 2 4 Even parity
*5 3 1 1 stop bit
*5 4 3  Flow Control - HW
5 5 1 Interface mode - SBI
6 1 4 Autoprint regardless of stability
6 2 1 Print stoppable
6 4 2 Tare after Print
Application F4=Tare <br>
Container Tare off <br>
Lineformat: GLP With ID code for data output (22 character format)
```

### Example: Scale menu settings for Sartorius Basic plus and LE, CP, GC, or GP series

The scales (e.g. Sartorius BP410) have a **TARE** key for zeroing and an **F** key for taring (if set up correctly). They do not support a **tare preset** natively, instead, it is emulated within the driver. If **tare preset** is activated, the scale display will nevertheless continue to show gross not net weight.
The settings are similar for other families.
Default settings are marked by an asterisk (*).
Use the **SBI4BP** scale driver.

```
9 9 1 Factory defaults
5 1 5 2400 Baud
* 7 Bit
5 2 4 Even parity
*5 3 1 1 stop bit
*5 4 2 Hardware handshake with 2 characters after CTS (use proper cable!)

6 1 4 Autoprint regardless of stability
6 2 2 Print not stoppable
6 4 1 Data output without automatic taring
7 1 2 Automatic output of the application parameters
7 2 2 With ID code for data output (22 character format)
7 3 2 Automatic output of tare memory data
2 1 6 Application program tare memory / net total
```

- Other scale families that use the Sartorius SBI protocol
  The available drivers are supported, not qualified.
  - Sartorius AC, IC, RC, MC series
    Use the **SBI4AC** scale driver.
  - Sartorius LP series
    Use the **SBI4LP** scale driver.

## Defining Scale Equipment Entities

Scale equipment entities are defined in Data Manager - Equipment. It is essential that the key data of a scale (ranges, precision, unit of measure, supported features), which is given in the technical manual of the scale, matches the data you provide in Data Manager - Equipment. For further information on how to create a new scale equipment entity in Data Manager - Equipment and the available scale properties, please refer to the "Data Manager User Documentation" [C2] (page 98).

- The **Terminal number** is typically **0** unless more than one scale is connected through the same terminal. In that case the **Terminal number** corresponds to the scale number **1, 2,** ... at the terminal.
- In order to use an Ethernet connection to the scale, the **Connection port** has to be left blank. Then you can provide the **TCP server address** and the **TCP port** of the scale.

The scale locale is set to **en_US** by default as required for the scale driver communication. There is no relation to the locale-related settings of the scale or of PharmaSuite.

> TIP
>
> The interface settings are pre-defined by the scale drivers.

## Testing Scale Commands

We recommend to test all scale commands with the **Get weight** phase of the Dispense package. The phase covers the three weighing-related features zeroing, taring, and weighing.
In Workflow Designer, build a workflow with the phase and execute the workflow in PharmaSuite for Production Execution.

## Troubleshooting Scale Communication

If the communication does not work properly, as a first step check the hardware setup and the settings at the terminal. For this purpose, shut down PharmaSuite for Production Execution and start Microsoft HyperTerminal (Windows XP: Programs | Accessories | Communication) or another Terminal program (e.g. freeware PuTTY tool).
Open a new connection on the respective COM port and set it up with the communication parameters defined for your connection. For an Ethernet-based communication, open the connection via TCP/IP Winsock to the respective Telnet server IP address and IP port of the scale.
A Terminal program allows you to establish a connection to the scale interface, to see the messages sent by the terminal, to type commands to be sent to the terminal, and to check if the terminal is responding in the expected way. Only if the test is successful, does it make sense to continue the test with PharmaSuite for Production Execution.

JAR-based scale drivers allow you to trace the communication sent to and received from the serial port. To activate trace logs, add the following lines to the *PC_MES\commons\base\config\log4j_ftps.properties* property file:

```
log4j.logger.com.rockwell.mes.services.eqm.ifc.scales=DEBUG
log4j.logger.com.rockwell.mes.services.eqm.ifc.scales=INFO
```

Typically, the result will be written to a log file in the following directory of the local client machine:
*c:\.FTPC\<MES-PS-HOST>\ProductionCentre\logs\PlantOpsClient\*

For a customized PharmaSuite system, proceed as follows:

1. Copy the *log4j_custom.properties* property file from the root folder in *custom-ssb-ftps.jar.jar* located at *c:\Users\<Windows User>\AppData\Roaming\Rockwell Automation\FactoryTalk ProductionCentre\jars\ShopOps\*.

2. To activate trace logs, add or modify the following lines:
   ```
   log4j.logger.com.rockwell.mes.services.eqm.ifc.scales=DEBUG
   log4j.logger.com.rockwell.mes.services.eqm.ifc.scales=INFO
   ```

3. Save the file as *log4j_custom_local.properties* to *c:\Users\<Windows user>\AppData\Roaming\Rockwell Automation\FactoryTalk ProductionCentre\logs\PlantOpsClient\* of the client machine on which you wish the traces to be enabled.

**TIP**

It is not required to restart PharmaSuite for Production Execution since the file is dynamically watched for changes every 10 seconds.

For more information, see chapter "Logging and Debugging" in Volume 4 of the "Technical Manual Configuration and Extension" [A8] (page 97).

### Troubleshooting Scale Communication in a Citrix Environment

If the scale communication is unreliable in a Citrix environment, we recommend to set the **LibraryHolder/services-eqm-ifc.jar/FilterNULBytes** configuration key to **true**. In case the issue persists for some scales, we recommend to set the **LibraryHolder/services-eqm-ifc.jar/RegisterSerialEventHandler** configuration key to **false**.

For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).

## Integrating New Label Printers

PharmaSuite supports printers with MS Windows-compatible drivers.

Check, if the manufacturer of the label printer delivers MS Windows drivers for it or if MS Windows itself supports the label printer by its own driver. Use the MS Windows hardware wizard to install the driver.

If the PharmaSuite application is not configured to display a print dialog for printer selection, the print job will be sent to the printer set as **Default Printer** in the Windows system. Therefore, it is essential that at least one printer is configured as a **Default Printer**.

To configure label printing in PharmaSuite, adapt the **Equipment/defaultLabelPrinter**, **Equipment/equipmentEntityLabel.showPrinterSelection**, **LabelDesign/defaultPrinter**, and **LabelDesign/simpleWeighingLabel.reprint.showPrinterSelection** configuration keys. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).

The size of the PharmaSuite product labels is 4 x 5 inches (width x height).

## Integrating New Barcode Scanners

PharmaSuite supports all keyboard-emulating barcode scanners, provided they can be configured to send a defined prefix and suffix.

Contrary to the integration of scales, there is no need for a specific device driver when using a barcode scanner. PharmaSuite expects that the MS Windows driver of the barcode scanner is able to emulate keyboard input. Moreover, it must be possible to configure the scanning device to send a defined prefix and suffix whenever a barcode is scanned. A prefix or suffix in this context is a keystroke that is sent before or after the data that has actually been scanned.

To make a scanner available for use with PharmaSuite, proceed as follows:

1. Connect the scanner to the computer.
   Usually, MS Windows will recognize the scanner as human interface device and install a suitable driver for it. For details, please refer to your scanner manual.
2. Configure the scanner prefix/suffix.
   Before you start configuring your scanner, we recommend to delete all currently configured scanner settings.
   In the default configuration, the prefix is the F4 function key, and the suffix is <CR> (carriage return). Very often you have to scan a specific barcode that is indicated in the scanner's symbology chart before scanning a prefix. For details on how to configure the prefix and suffix, please refer to your scanner manual.
   Prefix and suffix can be adjusted during system configuration. For further information, please refer to the PharmaSuite documentation. In particular, see chapter Configuration Keys of PharmaSuite in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).
   When using a 2D barcode scanner, you have to configure the Data Matrix symbology of your scanner.

   **TIP**
   It may be necessary to adjust the scanner's keyboard input language to the setting made in the Windows operating system to ensure that all special characters are transmitted correctly.

3. Test the scanner within the application.
   Scanning of a sublot barcode can be tested in PharmaSuite for Production Execution with the **Inventory Correction - Filter Sublots** pre-defined workflow step.
   Scanning of a batch barcode can be tested in PharmaSuite for Production Execution with the **Identify material** phase in the Dispense application.
   Scanning of a 2D barcode can be tested in PharmaSuite for Production Execution when executing orders or workflows.

## Creating Labels for Scales, Printers, and Containers

The Dispense application of PharmaSuite realizes a scanner-driven usability concept for scales, printers, and containers. Whenever during processing a communication with a scale, printer, or another piece of equipment is required, the operator can scan the barcode of the respective piece of equipment either to identify it or trigger actions (e.g. taring, printing).

The equipment barcodes printed on the labels must consist of the following data:

- Scale
  `<barcode of the scale as maintained in the basic data of the scale equipment entity>`
- Printer
  `!printer`
- Target (to close the target)
  `!closetarget`

# Best Practices for Managing User Accounts of a Client Operating System

This chapter provides best practices related to the management of user accounts of a client operating system in the context of terminal services. Please observe the instructions to establish an appropriate system configuration.

PharmaSuite clients are Java-based and run in a client operating system (e.g. MS Windows). A client operating system can be hosted directly on a physical machine (a device like a PC, tablet, scanner) or it can be hosted on a terminal services host. For this reason, a user has to log in to the client operating system first with his operating system-specific user account before accessing a PharmaSuite client.

FactoryTalk ProductionCentre supports various client configurations with different file location settings (see section "Changing the Default Configuration" in "Technical Manual Installation" [A7] (page 97)). If no FactoryTalk ProductionCentre download location has been configured, all client operating system users on the same machine share the same download directory (*c:\.FTPC*) to store their user-specific files, e.g.

- *~options.dat* keeps the station information or
- PharmaSuite client log files.

The following scenarios affect the user account configuration.

**Scenario 1**: Different client operating system users on one machine.

- Use case: With terminal services, the sessions of different users are hosted on the same host.
- Issue: All client operating system users on the same machine share the same file locations and files.
- Required change of installation configuration: Set the *clientHome* property to the user's profile directory to keep user-specific files separate.

**Scenario 2**: One client operating system user runs multiple PharmaSuite clients on different devices in different sessions.

- Use case: Devices need to be shared among different PharmaSuite users. This applies to mobile devices (e.g. scanners, tablets) that support an automatic generic system login. After the generic login, a terminal services session is automatically started to run a PharmaSuite client.
- Issue: All sessions share the same file locations and files despite the setting of the *clientHome* property.
- Recommend best practice: Configure an own device-specific user for each device. As a consequence, a terminal services session uses its own device-specific FactoryTalk ProductionCentre download location and each session can be mapped to its device and vice versa.

Chapter 5

# Managing Users and Access Rights

PS Administration is the PharmaSuite client for the administration of database objects, such as access privileges, lists, applications, users, or user groups.
For managing users and access rights for PharmaSuite, see "Implementation Guide PS Administration" [A1] (page 97).

# Exporting and Importing Master Recipes, Master Workflows, and Building Blocks

This section contains general information about the export and import of master recipes, master workflows, and building blocks, provides guidelines (page 38) how to perform exports and imports, instructions how to adapt the processes (page 46), and technical details (page 51).

Developing a master recipe or master workflow is an ambitious task and should be carefully planned. Thorough testing is essential before they can be approved. It is a good practice to manage their development life-cycle across multiple PharmaSuite instances (e.g. development system, test system, productive system). For this purpose, master recipes, master workflows, and custom building blocks need to be transferred from a source system to a target system.

The **PharmaSuite Export/Import Utility** tool exports and imports your master recipes, master workflows, and custom building blocks maintained with PharmaSuite Recipe and Workflow Designer.
For objects that are protected by an access privilege: The system only allows to export objects whose access privilege matches the access privilege of the logged-in user.

To export and import a master recipe, master workflow, or custom building block, perform the following steps:

1. On the source system, start the **PharmaSuite Export Utility** tool (page 39).
2. Provide the required configuration parameters and start the export.
3. The system displays the result of the export and stores a log file at the specified location.
   We recommend to check both.
4. Close the **PharmaSuite Export Utility** tool.
5. If required, copy the exported BML file (Batch Markup Language) to the location expected by the import.
6. On the target system, start the **PharmaSuite Import Utility** tool (page 42).
7. Provide the required configuration parameters and start the import.

8. The system displays the result of the import and stores a log file at the specified location.
   We recommend to check both.
9. Close the **PharmaSuite Import Utility** tool.

## Prerequisites

To enable a successful data transfer between two PharmaSuite systems, the following preconditions must be fulfilled:

- Both the source and target system must be of the same (maintenance) release. Examples: source = x.y and target = x.y is supported, source = x.y and target = x.y MRz is not supported.
- Any master data object (e.g. material/part, work center, station, access privilege, equipment class, equipment property type) that is referenced in the source system by the master recipe, master workflow, or custom building block to be exported must be available in the target system with the same attributes (e.g. identifier, description, version).
  This also applies also:
  - Equipment entities that are assigned to referenced equipment classes; therefore they are to be used during execution.
  - Equipment property types that are referenced by a relevant equipment class or entity.
- Any system building block and parameter class that is referenced by a master recipe, master workflow, or custom building block to be exported must have been deployed to the target system before they can be imported.
- Custom building blocks that have been used during recipe and workflow authoring are not required to be available in the target system. See "Deep copy vs. Shallow copy" in section "Technical Details" (page 51).
- The versions of the underlying B2MML schemas must be identical.
- The schema version of the BML files to be imported must match the expected schema version of the PharmaSuite release. Otherwise a warning will be logged.

## Transferred Data

The **PharmaSuite Export/Import Utility** tool exports and imports the following data:

- All data related to a master recipe, master workflow, or custom building block, including version, status, and MFC data.
  (Although BOM data is exported for convenience reasons, it is not imported but restored from the MFC-related data during the import process of a recipe with valid MFC data.)
- All data related to a custom building block, including version and status.
- Status transition and version history of the master recipe or master workflow.
  The data is exported to a B2MML file and imported into the **X_VersionHistory** application table.
  The data is cumulated if more than two PharmaSuite instances are used during the recipe or workflow life-cycle. Both status transition and version history of the current (local) system and the history data related to the exporting system are displayed in the corresponding dialog of Recipe and Workflow Designer.
- Checksum of the exported data.
  The import process uses a checksum to ensure that the transferred data is genuine. The checksum is not stored in the database.
  If the checksum is verified and does not match, the import is canceled to prevent the import of manipulated data.

## Working with the PharmaSuite Export/Import Utility

The **PharmaSuite Export/Import Utility** tool requires a set of configuration parameters specifying the export (page 39) and import (page 42), respectively. They can be provided in the following ways:

- In the **PharmaSuite Export/Import Utility** tool.
- In the *ExportConfiguration.xml* and *ImportConfiguration.xml* configuration files.
  The configuration files and their corresponding schema definition (*ExportConfiguration.xsd*, *ImportConfiguration.xsd*) can be found in the **apps-recipeb2mml-ifc.jar** FactoryTalk ProductionCentre **Library** object.
  Extract the files from the JAR file, adapt them according to your needs, and copy them to the locations specified with the *Configuration file* parameter in **PharmaSuite Export/Import Utility** tool.
  In the examples below, the red colored values must be adapted.

### Example of ExportConfiguration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Sample Recipe B2MML Export Configuration File -->
<FTPSMasterRecipeExport
  xmlns="http://www.rockwellsoftware.com/FTPS/B2MMlExportConfiguration"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.rockwellsoftware.com/FTPS/B2MMlExportConfiguration
                     ../config/ExportConfiguration.xsd ">
  <SourceRecipeElement
    recipeElementType="Master Recipe"
    recipeElementName="MYC First Recipe"
    recipeElementVersion="2" />
  <SourceFTPSInstance URL="jnp://SourceAppServer:1099"
    username="MYC login" password="MYC password" />
  <OutputFiles outputDirectory="C:\MYC exports"
    recipeOutputFilename="MYC First Recipe.bml"
    configOutputFilename="MYC First Recipe.xml"
    logOutputFilename="MYC First Recipe.log" />
</FTPSMasterRecipeExport>
```

### Example of ImportConfiguration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Sample Recipe B2MML Import Configuration File -->
<FTPSMasterRecipeImport
  xmlns="http://www.rockwellsoftware.com/FTPS/B2MMLImportConfiguration"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.rockwellsoftware.com/FTPS/B2MMLImportConfiguration
                     ../config/ImportConfiguration.xsd ">
  <TargetFTPSInstance URL="jnp://TargetAppServer:1099"
    username="MYC login" password="MYC password" />
  <InputFile inputDirectory="C:\MYC imports" recipeInputFilename="MYC First Recipe.bml"
                            ignoreXMLSignature="false" />
  <TargetRecipeElement targetType="Master Recipe" />
</FTPSMasterRecipeImport>
```

**TIP**

Parameters specified in the **PharmaSuite Export/Import Utility** tool override the corresponding specification of the configuration file.
If a parameter is not specified, the corresponding value from the configuration file is taken.

## Exporting a Master Recipe, Master Workflow, or Custom Building Block

To export a master recipe, master workflow, or custom building block, proceed as follows:

1. In Process Designer on the source system, run the **FTPSExportUtility** form to start the **PharmaSuite Export Utility** tool.
   OR
   In Shop Operations on the source system, open the **FTPSExportUtility** form to start the **PharmaSuite Export Utility** tool.
2. Provide the required configuration parameters.
   - Configuration file
     Optional, not available in the *ExportConfiguration.xml* file.
     Path to the export configuration file.
   - Component type
     *recipeElementType* in the *ExportConfiguration.xml* file.
     Type of the component to be exported (e.g. Master recipe, Master workflow, Phase).
   - Component ID
     *recipeElementName* in the *ExportConfiguration.xml* file.
     Identifier of the entity to be exported.

- Version or revision
  *recipeElementVersion* in the *ExportConfiguration.xml* file.
  Version or revision of the entity to be exported.
- URL
  Only available in the *ExportConfiguration.xml* file.
  IIOP address of the PharmaSuite server.
  Not supported in the current release.
- username
  Only available in the *ExportConfiguration.xml* file.
  Name to log in to the PharmaSuite server.
  Not supported in the current release.
- password
  Only available in the *ExportConfiguration.xml* file.
  Password to log in to the PharmaSuite server.
  Not supported in the current release.
- Output directory
  *outputDirectory* in the *ExportConfiguration.xml* file.
  Target location (on the local machine) where to save the exported file and the log file. The default is the user's *My Documents* directory.
- Output file
  *recipeOutputFilename* in the *ExportConfiguration.xml* file.
  Name of the exported file. The default file name is
  *<recipeElementName>_<recipeElementVersion>.bml*.
  Existing files with this name will be overwritten during export.
- Configuration file template for import
  *configOutputFilename* in the *ExportConfiguration.xml* file.
  Configuration file that can be used as a template for controlling the import of the exported master recipe, master workflow, or custom building block into a target PharmaSuite system
  Not supported in the current release.
- Log file
  *logOutputFilename* in the *ExportConfiguration.xml* file.
  Name of the log file.
  Existing files with this name will be overwritten during export.

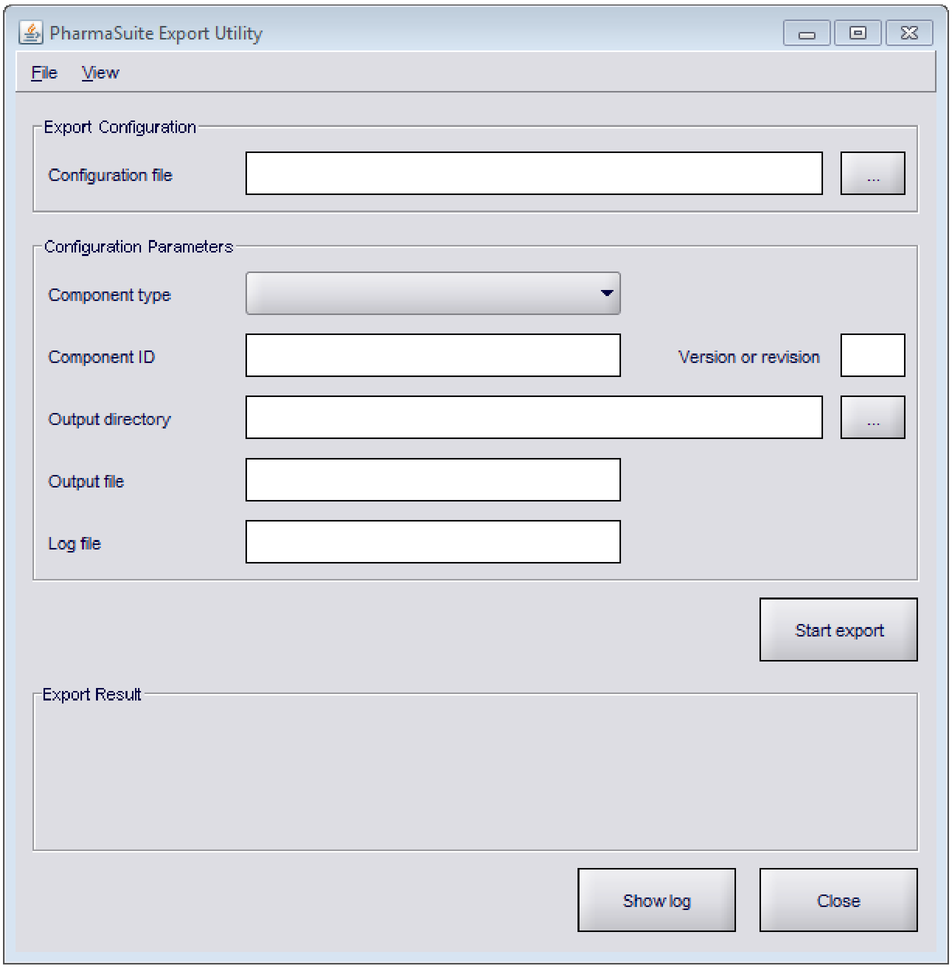


*Figure 1: PharmaSuite Export Utility tool*

3. Click the **Start export** button.

4. The system displays a summary of the export process in the *Export Result* panel; details are stored in the log file.
   We recommend to check both.
   If the export was successful, the system stores a B2MML file (Business To Manufacturing Markup Language) at the specified location. Continue with the next step.
   If the export was not successful, resolve the issues listed in the log file and repeat the export. For more information, see section "Troubleshooting" (page 44).

5. Close the **PharmaSuite Export Utility** tool.

### Importing a Master Recipe, Master Workflow, or Custom Building Block

To import a master recipe, master workflow, or custom building block, proceed as follows:

1. Make sure, the exported file is accessible from the target machine. If required, copy the file to the location expected by the import.
2. In Process Designer on the target system, run the **FTPSImportUtility** form to start the **PharmaSuite Import Utility** tool.  
   OR  
   In Shop Operations on the target system, open the **FTPSImportUtility** form to start the **PharmaSuite Import Utility** tool.
3. Provide the required configuration parameters.
   - Configuration file  
     Optional, not available in the *ImportConfiguration.xml* file.  
     Path to the import configuration file.
   - Component type  
     *recipeElementType* in the *InportConfiguration.xml* file.  
     Type of the component to be imported (e.g. Master recipe, Master workflow, Phase).
   - URL  
     Only available in the *ImportConfiguration.xml* file.  
     IIOP address of the PharmaSuite server.  
     Not supported in the current release.
   - username  
     Only available in the *ImportConfiguration.xml* file.  
     Name to log in to the PharmaSuite server.  
     Not supported in the current release.
   - password  
     Only available in the *ImportConfiguration.xml* file.  
     Password to log in to the PharmaSuite server.  
     Not supported in the current release.
   - Input file  
     *recipeInputFilename* and *inputDirectory* in the *ImportConfiguration.xml* file.  
     Path and name of the exported file. The default is the user's *My Documents* directory.
   - Log directory  
     *outputDirectory* in the *ImportConfiguration.xml* file.  
     Target location where to save the log file. The default is the user's *My Documents* directory.

- Log file
  *logOutputFilename* in the *ImportConfiguration.xml* file.
  Name of the log file.
  Existing files with this name will be overwritten during import.
- Name Maps
  Only available in the *ImportConfiguration.xml* file.
  Mapping rules to map object names between the source and target system.
  Not supported in the current release.

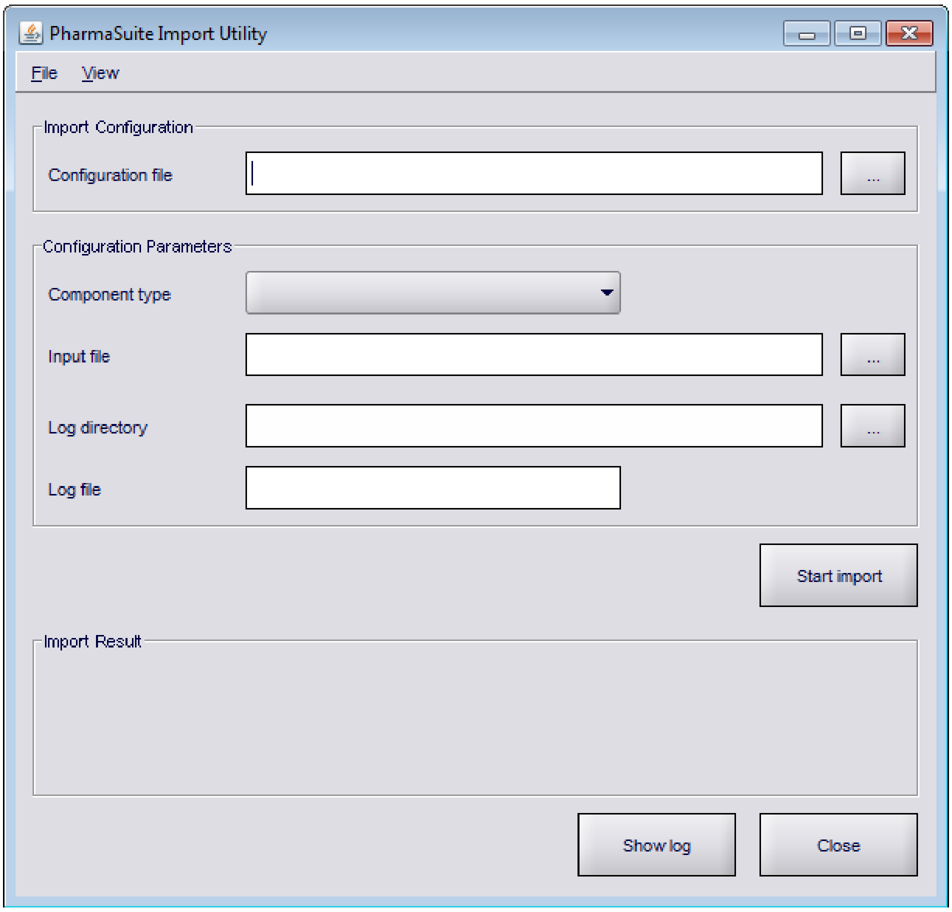


*Figure 2: PharmaSuite Import Utility tool*

4. Click the **Start import** button.
5. The system displays a summary of the import process in the *Import Result* panel; details are stored in the log file.
   We recommend to check both.
   If the import was successful, the system stores the master recipe, master workflow, or custom building block in the database. The imported object is ready for use in the target system. Continue with the next step.
   If the import was not successful, resolve the issues listed in the log file. Depending on the issue, you must repeat the import, update data in the Production Management Client or Recipe and Workflow Designer on the source or target system, or repeat the whole process. For more information, see section "Troubleshooting" (page 44).
6. Close the **PharmaSuite Import Utility** tool.

## Troubleshooting

The following issues may occur during the export or import process.

Export and import:

- **Cause**: Incomplete configuration
  Required configuration parameters are not specified or invalid.
  - **Effect**: The process is aborted.
  - **Action**: Update the configuration parameters.
- **Cause**: File or path does not exist
  Files or paths specified for the export or import do not exist.
  - **Effect**: The process is aborted.
  - **Action**: Update the configuration parameters and/or create the directories.

Only export:

- **Cause**: Master recipe, master workflow, or custom building block not found
  Master recipe, master workflow, or custom building block specified for the export cannot be found.
  - **Effect**: The process is aborted.
  - **Action**: Update the configuration parameters and/or create the master recipe, master workflow, or custom building block.

Only import:

- **Cause**: Checksum validation failed  
  The exported file has been edited manually.
  - **Effect**: The process is aborted.
  - **Action**: Make sure that the file to be imported has not been edited.
- **Cause**: Referenced object does not exist  
  An object referenced by an exported master recipe, master workflow, or custom building block cannot be found in the target database.
  - **Effect**: The process continues and logs the issue in the log file.  
    The imported data is not made persistent in the database. It will be deleted when the **PharmaSuite Import Utility** tool is closed.
  - **Action**: Close the **PharmaSuite Import Utility** tool. Make the missing object available in the target database or update the master recipe, master workflow, or custom building block to be exported. Repeat the export and import process.

  > TIP  
  > The import fails if a work center assignment parameter references a work center or station object that does not exist in the target system. Under specific circumstances, it may become necessary that the work center assignment parameter is ignored during import. This behavior can be enabled with the **ImportRecipeBuildingBlockIgnoreNonExistingEquipment** configuration key. For details, see "Configuration Keys for Recipe and Workflow Designer", chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).  
  > If a work center assignment parameter is not added due to a missing reference and the exported master recipe or master workflow was in the **Verification** or **Valid** status, the imported master recipe or master workflow is in the **Edit** status. Custom building blocks in the **Approved** or **Verification** statuses are in the **Draft** status after the import. Thus, they can be updated manually and re-validated.

- **Cause**: Master recipe, master workflow, or custom building block already exists  
  The object to be imported already exists in the target database with the same identifier and version/revision.
  - **Effect**: The process is aborted and logs the issue in the log file.
  - **Action**: The action depends on the data, for example, overlapping periods of validity prevent the import.
- **Cause**: Recipe and Workflow Designer checks failed  
  Along with the import process, the specified checks for Recipe and Workflow Designer are executed. One or more checks failed.

- **Effect**: The imported data is stored in the database and can be updated.
  If the exported master recipe or master workflow was in the **Verification** or **Valid** status, the imported master recipe or master workflow is in the **Edit** status. Thus, they can be corrected manually and re-validated or deleted.
- **Action**: Open the imported master recipe, master workflow, or custom building block in Recipe and Workflow Designer and review the issue(s) displayed in the Messages window.

## Adapting the PharmaSuite Export/Import Utility

The **PharmaSuite Export/Import Utility** tool can be adapted in the following areas:

- checks to be executed by Recipe and Workflow Designer (page 46),
- customer-specific application tables used as parameter attributes (page 47),
- customer-specific parameter attributes (page 49), e.g. material, equipment requirement, or privilege parameters, and
- customer-specific UDAs of a master recipe (page 50).

### Checks of Recipe and Workflow Designer

During the import process, Recipe and Workflow Designer performs several checks. They can be adapted according to the description in chapter "Modifying and Adding Checks" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97).

## Customer-specific Application Tables Used as Parameter Attributes

The availability of referenced objects in both the source and target system is a prerequisite (page 36). This ensures that only a reference that can uniquely identify the object is exported. During the import process, the tool checks whether the object already exists in the target system. If so, it creates a matching reference.
It is obvious that the key attribute of the object cannot be used as such a reference since it will be very likely different in both systems.

Application table rows are keyed objects with a unique **name** attribute. Often this key is a system-controlled UUID and thus not necessarily identical for the same object in the source and target system. Only if the **name** attribute is identical in both systems, can an import into another PharmaSuite instance work.
Otherwise and if, for example, the external-speaking unique key requires more than one attribute, the **PharmaSuite Export/Import Utility** tool can be extended.

Example: A process parameter attribute refers to the **X_GHSStatement** object of PharmaSuite. The object's name is a UUID, thus an external-speaking unique key has to be found: **X_ghsIdentifier**.
The extension of the **PharmaSuite Export/Import Utility** tool requires the following tasks:

1. Add support to an additional ATRow (here: **X_GHSStatement**), see *hasATRowReferenceConverter* method below.
2. Add a method to convert the object to an external-speaking key (here: **String**), see *atRowReference2String* method below.
3. Add a method to retrieve the object from an external-speaking key (here: **String**), see *createATRowFilter4AtRowReference* method below.

For this purpose, provide enhanced service classes for the services of the **PharmaSuite Export/Import Utility** tool.

### Example: Adapted export service

```
public class ExtendedBatchParameterConverter extends BatchParameterConverter {
  @Override
  protected String atRowReference2String(ATRow atRow) {
    String name = atRow.getATDefinition().getName();
    if (name.equals("X_GHSStatement")) {
      return (String) atRow.getValue("X_ghsIdentifier");
    } else {
      return super.atRowReference2String(atRow);
    }
  }

  @Override
  public boolean hasATRowReferenceConverter(ATDefinition atDef) {
    return atDef.getName().equals("X_GHSStatement") ||
                                   super.hasATRowReferenceConverter(atDef);
  }
}
```

### Example: Adapted import service for master recipes/master workflows

```
public class ExtendedMasterRecipeImportBuilder extends MasterRecipeImportBuilder {

  @Override
  public ATRowFilter createATRowFilter4AtRowReference(String atRowReferenceString,
         ATDefinition applicationTable,
         ValueStringType valueStringType) throws DatasweepException {
    ATRowFilter atRowFilter = new ATRowFilter(applicationTable.getName(),
                                  PCContext.getServerImpl());
    if (applicationTable.getName().equals("X_GHSStatement")) {
      atRowFilter =
               atRowFilter.forColumnNameEqualTo("X_ghsIdentifier",
atRowReferenceString);
    } else {
      return super.createATRowFilter4AtRowReference(atRowReferenceString,
                                  applicationTable, valueStringType);
    }
    return atRowFilter;
  }
}
```

**TIP**

Master workflows use the same import classes as master recipes. To distinguish between master recipes and master workflows apply one of the following options:

- use the *processingType* attribute of the *IMESMasterRecipe* interface or
- overwrite one of the type-related methods (*extractMasterWorkflowAttributes* and *extractMasterRecipeAttributes*) of the *IMasterRecipeImportBuilder* interface.

Register the extended classes by means of the PharmaSuite standard *serviceFactory* mechanism; then

- the *exporter* will create appropriate references for **X_GHSStatement** object in the BML file (Batch Markup Language) of the process parameter and
- during import, the *importer* will check for the existence of **X_GHSStatement** objects and, if applicable, create appropriate references.

## Customer-specific Parameter Attributes

Customer-specific parameter attributes are, for example, material, equipment requirement, or privilege parameters.

If you have added customer-specific attributes to your custom building blocks, ensure that the AT definitions of the respective parameters in both the source and target systems have exactly the same customer-specific attributes.
An attribute name must start with a prefix. According to the naming conventions (page 3), **X_** and **RS_** are reserved for PharmaSuite. This ensures that the attribute is recognized as customer-specific and exported and imported automatically.

### Customer-specific Parameter Attributes with References

The **PharmaSuite Export/Import Utility** tool works for all parameter attributes of simple data types such as String, Boolean, Long, MeasuredValue, and Decimal. For references to other database tables, however, you need to distinguish the following two scenarios:

- If the **name** column is a unique column, the **PharmaSuite Export/Import Utility** tool is supported by default. This also applies to all standard objects, such as materials or batches.
- If the **name** column is not unique or not human-readable, the **PharmaSuite Export/Import Utility** tool needs to be extended as described in the following steps. Typically, this is necessary for AT definitions.

  1. Create an instance of the *IReferenceManager* interface for the wrapper class of the custom AT definition:

     ```
     class CustomerClassReferenceManager implements
     IReferenceManager<CustomerClass> {
     [...]
     }
     ```

     The class must implement the following methods:

     - *String getUniqueIdentifier(CustomerClass object)*
     - *CustomerClass getObjectByUniqueIdentifier(String identifier)*
     - *Keyed getKeyedByUniqueIdentifier(String identifier)*

2. Register an instance of the class in the central *PhaseParameterAttributeReferenceRegistry* registry:

   ```
   PhaseParameterAttributeReferenceRegistry.INSTANCE.
     registerReferenceManager(CustomerClass.ATDEFINITION_NAME,
     new CustomerClass());
   ```

   This manager instance enables

   - the export service to write the unique identifier of the object into the BML file and
   - the import service to read the correct record from the database and convert it into a wrapper object.

## Customer-specific UDAs of a Master Recipe

If you have added customer-specific UDAs to master recipes, ensure that the UDAs are available in both the source and target systems and have the same data type.
An attribute name must start with a prefix. According to the naming conventions (page 3), **X_** and **RS_** are reserved for PharmaSuite. This ensures that the attribute is recognized as customer-specific and exported and imported automatically.

## Technical Details

When exporting and importing master recipes, master workflows, and custom building blocks, you should be familiar with the following technical details:

- Format of the exported file
  The format is based on the S88 industry standard. The required components have been adapted to PharmaSuite-specific needs. Thus, the data is stored in a B2MML XML structure.
- Object locking
  Object locking is not supported. Even if an object is locked for editing by another user, the object can be exported.

  > TIP
  > Please note that the object can therefore be in an inconsistent or incomplete state.

- Deep Copy vs. Shallow Copy
  The **PharmaSuite Export/Import Utility** tool follows the Shallow Copy concept. Consider a master recipe, master workflow, or custom building block built of other custom building blocks. The information which custom building blocks are used is stored in the database, displayed in the property window of Recipe and Workflow Designer, and transferred to the target system. The custom building blocks themselves are not transferred, since there is no Deep Copy.
  If the custom building blocks are needed in the target system, each custom building block must be exported and imported by means of the **PharmaSuite Export/Import Utility** tool.

Chapter 7

# Importing Equipment Classes and (Template) Entities from an External Source

The default functions for importing equipment classes and (template) entities are available in Data Manager. They reject an import file and abort the import if the checksum of a file has been changed.

> **TIP**
>
> For the tag and point import, a checksum verification is not supported.

If you need to import an equipment class or (template) entity from an external source without a checksum, use the *IS88EquipmentExportImportService.importEquipment(IEquipmentImportConfigurationBuilder)* method. The *importIgnoreChecksum(boolean)* setter in the *IEquipmentImportConfigurationBuilder* interface can be used in order to ignore a signature, i.e. an XML file can be imported without having a checksum defined.

Chapter 8

# Managing Device Identifiers

This chapter describes how a device identifier is generated and how to manage its format depending on the plant infrastructure where the device is located. PharmaSuite uses a device's identifier during execution with the Production Execution Client and displays it in the Cockpit.

The unique default device identifier fits with typical usage scenarios and correlates with

- the computer name (or host name) if a user runs a PharmaSuite application on a physical device directly or
- the client name (or identifier of a remote client) if a user runs a PharmaSuite application using terminal services.

The device identifier is always determined when the Production Execution Client is started. If you have to reuse a terminal server session from another hardware client, you must restart the Production Execution Client in order to get a unique device identifier.

> TIP
>
> It is assumed that the plant infrastructure guarantees that the default source of the device identifier is set and managed to be unique and stable.

## What Are the Computer and Client Names

The **computer name** is a unique name in a workgroup/domain used to identify the physical machine on a network. However, the computer name cannot be used in all cases. For instance, it cannot be used if an application runs on a terminal server and the computer name is displayed on another physical device. In this case, all applications are hosted by the server, so the computer name is always the same and is equal to the server's computer name. However, the **client name** represents the computer name of the physical device accessing the server. Therefore the client name is used as device identifier.

When Terminal Services is used it can easily be recognized if an application runs on a server. For this purpose, check if the SESSIONNAME environment variable exists and is not **Console**. This variable does not exist by default; it is initially created when a computer is accessed remotely and the session is initialized by Terminal Services. If the remote connection is closed and the computer is accessed locally, the variable is reset to **Console**.

## Configuring the Device Identifier Pattern for Terminal Services

The pattern of the device identifier is defined with the **commons-base-ifc.jar/DeviceIdentificationPatternTerminalServices** configuration key (see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97)).

By default, when using terminal services, the device identifier correlates to the CLIENTNAME environment variable.

If the default pattern cannot be used to generate a unique device identifier, adapt the value of the configuration key. We recommend to use a combination of the following environment variables:

- COMPUTERNAME: Name of the computer where an application runs.
- SESSIONNAME: When a computer is accessed remotely, the connection is identified by a session; the session name is unique in the context of the computer.
- CLIENTNAME: When a computer is accessed remotely, the client is identified by a client name; name of the client computer.
- USERNAME: Name of the user corresponding to the current account.

TIP

Verify the generated device identifier in the **About** dialog of your PharmaSuite client and the actual values of the related environment variables in the **Installation Details** dialog.

If this configuration approach is not sufficient to provide a unique and stable device identifier, overwrite the *com.rockwell.mes.commons.base.ifc.objectlock.IDeviceIdentificationService* interface with a custom implementation of the generation mechanism.

## Example Configuration

In the example configuration, two terminal servers are used to run the PharmaSuite applications. Different physical devices are connected to these two servers. Additionally, PharmaSuite runs on a physical device in a local session.

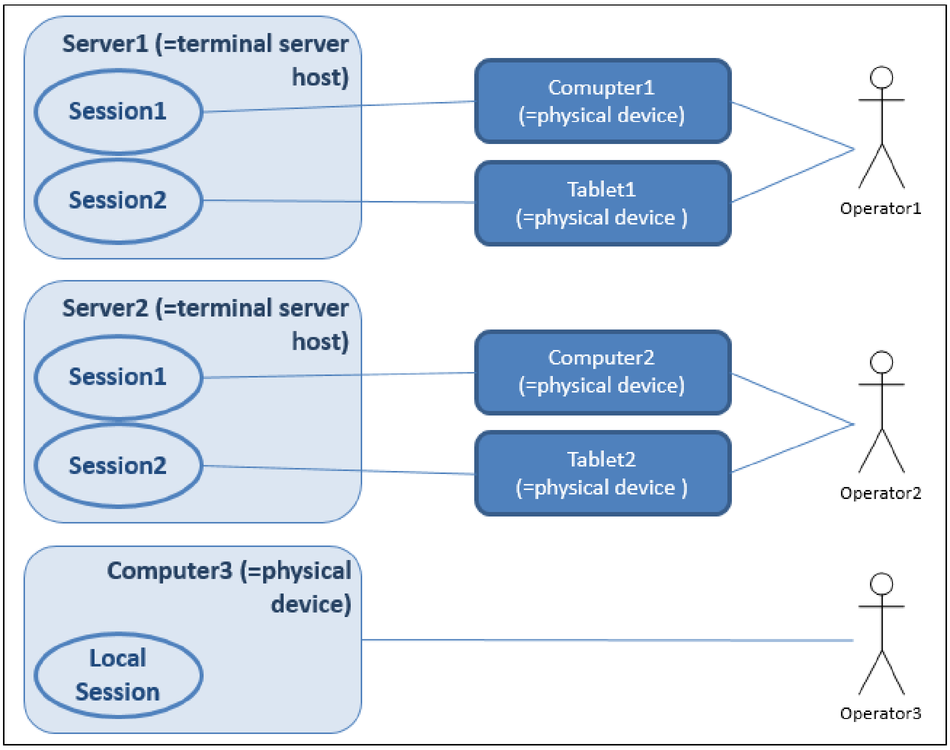

Server1 (=terminal server host)
Session1
Session2
Comuputer1 (=physical device)
Tablet1 (=physical device )
Operator1
Server2 (=terminal server host)
Session1
Session2
Computer2 (=physical device)
Tablet2 (=physical device )
Operator2
Computer3 (=physical device)
Local Session
Operator3

*Figure 3: Example configuration*

The following table shows the result of the default device identifier generation for the example configuration:

| Physical device (accessed/used by operator) | Device identifier (Generated) | Environment variables | | | |
|---|---|---|---|---|---|
| | | COMPUTERNAME | SESSIONNAME | CLIENTNAME | USERNAME |
| Computer1 | Computer1 | Server1 | Session1 | Computer1 | Operator1 |
| Tablet1 | Tablet1 | Server1 | Session2 | Tablet1 | Operator1 |
| Computer2 | Computer2 | Server2 | Session1 | Computer2 | Operator2 |
| Tablet2 | Tablet2 | Server2 | Session2 | Tablet2 | Operator2 |
| Computer3 | Computer3 | Computer3 | N/A or "Console" | N/A | Operator3 |

Chapter 9

# Localizing Each as Unit of Measure

This section describes the meaning of the **each** unit of measure within PharmaSuite.

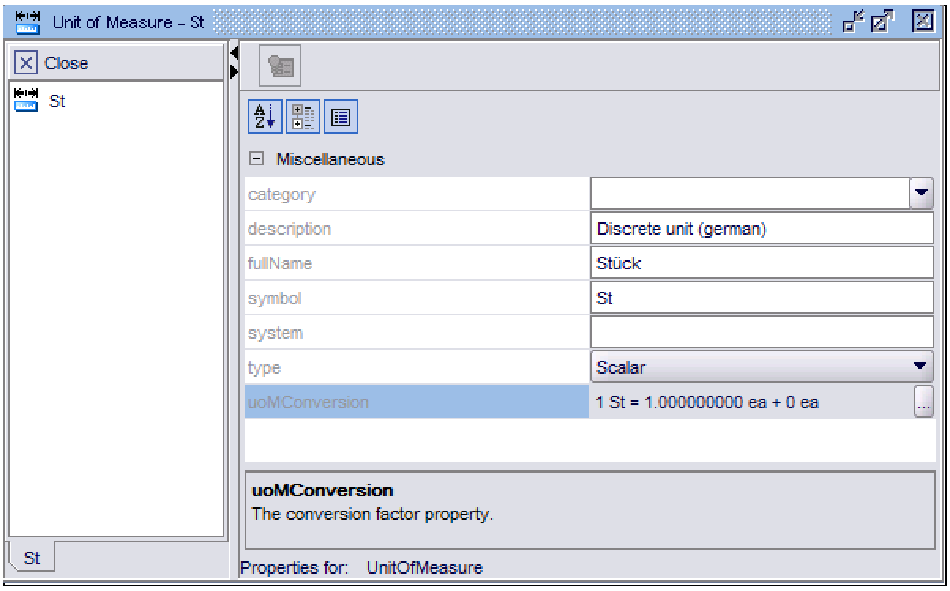
In case the unit of measure of the produced material must be a discrete unit of measure, the **each** (**ea**) unit of measure must be used. That means, that the produced material is measured either in **ea** or a unit that is convertible into **ea**.

For information on managing units of measure, please refer to the FactoryTalk ProductionCentre documentation. In particular, see section "Units of Measures" in "Process Designer Online Help" [B1] (page 97).

## Adding a Localized Discrete Unit of Measure

To create and configure a discrete unit of measure that is convertible into the **ea** unit of measure, proceed as follows:

1. In Process Designer, right-click the **Unit of Measures** node and select the **New Unit of Measure** function.
2. Define the properties as required (e.g. **description**, **full name**, **symbol**).
3. In the **type** option list, select **Scalar**.
4. In the **uomConversion** text box type **1.00 to UoM ea** as conversion factor.



| | |
|---|---|
| category | |
| description | Discrete unit (german) |
| fullName | Stück |
| symbol | St |
| system | |
| type | Scalar |
| uoMConversion | 1 St = 1.000000000 ea + 0 ea |

**uoMConversion**
The conversion factor property.

*Figure 4: German localization of each*

5. Save and close the new unit of measure.

Chapter 10

# Using Event Sheets in PharmaSuite

PharmaSuite uses the concept of event sheets to allow the user to monitor status changes, to execute activity sets, and to run servers. For more information on event sheets, please refer to sections "Using Event Sheets" and "Monitoring the Event Sheet" in "FactoryTalk ProductionCentre Plant Operations Release 10.4 Server Installation Guide - JBoss Advanced" [A5] (page 97) or "FactoryTalk ProductionCentre Plant Operations Release 10.4 Server Installation Guide - JBoss Stand-Alone" [A6] (page 97).

For the JMS communication of the PharmaSuite clients with the Transition server, the EBR server, the Automation Integration server, the Triggered Operation Management server, and the Operation Execution server, PharmaSuite employs the widely used open source messaging server Apache ActiveMQ. The ActiveMQ Java process is typically run as a Windows service. It is the prerequisite for the usage of the **PharmaSuite_Transition_Server**, **PharmaSuite_EBR_Server**, **PharmaSuite_AI_Server**, **PharmaSuite_TOM_Server**, and **PharmaSuite_OE_Server** event sheets.

The following event sheets are available:

- **PharmaSuite_Transition_Server** checks the period of validity of master recipes, master workflows, and batches and sets the status accordingly.
  The execution frequency for each task is encoded in the **cronExpression** property of the corresponding **CalendarEvents** container. For details, see sections "Event Sheets" and "CalendarEvents Container" in "Process Designer Online Help" [B1] (page 97).
  - Master recipe, master workflow (*VersionStatusTransition*): status transition from **Scheduled** to **Valid** and from **Valid** to **Archived**.
  - Batch (*BatchStatusUpdate*): performs the configured transition in case of expiration. For details of the **BatchQuality** flexible state model, see chapter "Configuring Flexible State Models" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97).

In addition, the event sheet is responsible for the following tasks:

- It deletes obsolete procedures of S88 master recipes and master workflows from the database (*deleteObsoleteExpandedReps* calendar event container). When a master recipe or master workflow is deleted in Recipe and Workflow Designer, the procedure (and all its dependent entities, parameters, and activity sets) are not deleted along with it immediately as this process can be very time-consuming. The procedure is only detached and marked for deletion. The event sheet triggers a service call that performs the actual deletion of those procedures.
- It deletes obsolete procedures of S88 control recipes and workflows from the database (*deleteDetachedS88ControlRecipeStructures* calendar event container). When an operation of an exploded S88 order or workflow is reset in the Production Management Client, the procedure (with all its dependent entities and parameters) is not deleted immediately as this process can be very time-consuming. The procedure is only detached and marked for deletion. The event sheet triggers a service call that performs the actual deletion of those procedures.
- In case external exception handling is configured, the event sheet listens for messages from an external QMS (Quality Management System) and processes the messages accordingly.
- In case PharmaSuite is configured to communicate with Warehouse Management, the event sheet listens for messages from Warehouse Management and processes the messages accordingly.
- For external equipment management, the event sheet listens for equipment management messages from an external system and processes the messages accordingly.
- For equipment entities whose expiry date has elapsed, it enforces a transition according to the assigned equipment transition. The execution frequency is not configurable with a CalenderEvent of Process Designer. It is defined with the **LibraryHolder/services-s88-ifc.jar/expiryDateTransitionCheckInterval** configuration key. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).
  The complete handling is encapsulated in the single-instance *S88EquipmentEntityExpiryDateTransitionEnforcer* object, which is created in the event sheet context. The system tracks specific data of skipped and failed status transitions of equipment entities. For more details, see chapter "Documenting Skipped and Failed Automatic Transitions of Equipment Entities" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97).

- **PharmaSuite_EBR_Server**, **PharmaSuite_AI_Server**, **PharmaSuite_TOM_Server**, and **PharmaSuite_OE_Server**
  - **PharmaSuite_EBR_Server** executes activity sets of S88 procedures and unit procedures related to the execution of PharmaSuite control recipes and workflows. In addition, it is responsible for processing incoming messages from a Distributed Control System, in case a DCS is configured to communicate with PharmaSuite.
  - **PharmaSuite_AI_Server** runs the Automation Integration server and communicates with FactoryTalk Live Data.
  - **PharmaSuite_TOM_Server** (Triggered Operation Management server) communicates with the **PharmaSuite_OE_Server** event sheet and manages the states of ETOs (Event-triggered operation).
  - **PharmaSuite_OE_Server** runs the Operation Execution server and supports the execution of EBR operations and phases having no user interface. EBR operations and phases with a user interface are visible in the Production Execution Client.

Chapter 11

# Monitoring PharmaSuite and Related Components

This chapter describes how to monitor a PharmaSuite system and related components based on a JBoss application server. Monitoring can be done for the following components:

- FactoryTalk ProductionCentre servers (page 66)
- PharmaSuite services (page 68)
- PharmaSuite clients (page 94)

PharmaSuite provides additional details with its **About** dialog (page 65), the **Administrator Console** tool (OE server operations (page 84), EBR server procedures (page 74), communication connection to OE and EBR server (page 90)), and the **Automation Integration Access Verification** tool (page 90).

For further information, please refer to the following documentation:

- Database-specific information:
  "FactoryTalk ProductionCentre Administrator Release 10.4 User's Guide" [A7] (page 97)
- Monitoring and usage of JDK tools
  Troubleshooting Guide for HotSpot VM [D4] (page 98)
- Monitoring with JConsole
  Using JConsole - Java SE Monitoring and Management Guide [D5] (page 98)

## PharmaSuite's About Dialog

The **About** dialog provides general information about the current application (e.g. version, logged-in user, memory consumption) and installation details (e.g. application server, application configuration).
Use this information for monitoring your PharmaSuite installation and the statuses of PharmaSuite services (e.g. AI server, EBR server, ActiveMQ broker).

> **TIP**
>
> For support cases, please provide the displayed installation details for analysis purposes.

## Monitoring FactoryTalk ProductionCentre Servers

The following table describes resources to be monitored for various server types. Please note that the given metrics should be considered as basic guidelines. They do not represent a comprehensive list of all circumstances that need to be addressed to prevent production issues.
The code can be mapped to various server types (see below).

| Code | Description | Boundary | Action |
|---|---|---|---|
| A | **Monitor CPUs for sustained utilization over 80 %**. On multicore systems even one CPU having high load may indicate a potential problem or runaway process. | **> 80 %** for one CPU | Identify the process causing the issue. |
| B | **Monitor** for available **memory < 5 %** of total memory. Running out of memory may cause poor performance or a server to hang. | Available memory **< 5 %** | Identify the process causing the issue. |
| C | **Monitor** for adequate **network quality, bandwidth, and latency**. Low bandwidth or high latency indicates network congestion and may cause poor performance.<br>TIP<br>Network packets, bandwidth, and latency can be monitored with Windows Performance Monitor.<br>There should be next to **0** errors and discarded packets. | Bandwidth:<br>**< 100 Mb/s**<br>Latency: **> 20 ms** | Analyze your network and resolve the issue. |
| D | **Monitor disk queue length for queues longer than 2**. If the queue, per physical disk, is longer than 2, this may indicate that your disks may be a performance bottleneck.<br>TIP<br>Disk queues can be monitored with Windows Performance Monitor. | Queues per physical disk longer than **2**. | Identify the process causing the issue.<br>Check if the corresponding component has performance issues.<br>If so, check the configuration of the component and/or consider to use hardware with better performance. |
| E | **Monitor** the **JBoss [EAP] service**. If this service is not running, FactoryTalk ProductionCentre is not running. | Service **stopped**. | Restart the JBoss and SOS services. |
| F | **Monitor** the **JBoss HTTP port**.<br>E.g. with telnet. | **No response** from the port. | Restart the JBoss and SOS services. |
| G | **Monitor** the **JBoss IIOP port**.<br>E.g. with telnet. | **No response** from the port. | Restart the JBoss and SOS services. |
| H | **Monitor JBoss logs**.<br>Be familiar with your logs; which messages are routine and which are not. | Error messages. | Depends on error message. |

| Code | Description | Boundary | Action |
|---|---|---|---|
| I | **Monitor** the available **Java Heap space for JBoss**, running out of heap space is equivalent to running out of free memory. | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the JBoss and SOS services.<br>Consider to adapt the maximum heap size. |
| J | **Monitor** the **Rockwell Shop Operations Server (SOS) service(s)**. If this service(s) is(are) not running, SOS is not running. | Service **stopped**. | Restart the SOS service(s). |
| K | **Monitor SOS log files**.<br>Be familiar with your logs; which messages are routine and which are not. | Error messages. | Depends on error message. |
| L | **Monitor** the **SOS Jetty ports**:<br>**8084** (PharmaSuite Transition server),<br>**8085** (PharmaSuite EBR server),<br>**8086** (PharmaSuite AI server),<br>**8087** (PharmaSuite TOM server), and<br>**8088** (PharmaSuite OE server).<br>These are the defaults port for SOS.<br>TIP<br>The SOS Jetty ports defined at installation in *ShopOperationsServer.xml* must be used. | **No response** from the port. | Restart the SOS service(s). |
| M | **Monitor** the available **heap space for Shop Operations Server(s)**. SOS runs under a JVM just like JBoss. | See sections "Monitoring PharmaSuite Services" (page 68) and "Monitoring PharmaSuite Clients" (page 94). | Restart the SOS service(s).<br>Consider to adapt the maximum heap size. |
| N | **Monitor** the **disk space**.<br>Running out of free disk space may cause data loss and computer hangs. | Free disk space **< 10 %** | Free disk space. |
| O | Offending or runaway processes. These processes may use up system resources. | Depends on the process. | Depends on the process.<br>It may have to be shut down. |
| P | **System Event Log**. It contains messages from the OS and other applications (e.g. SQL Server). | Warning or error messages. | Depends on message. |

| Code | Description | Boundary | Action |
|---|---|---|---|
| Q | **Monitor** the **number of connections** per used ports. | **> 1,000** | Identify the process causing the issue. Fix the connection leak inside the component. |

| Server type | Codes of monitor resources |
|---|---|
| JBoss Application server | A, B, C, D, E, F, G, H, I, N, O, P, Q |
| Rockwell Software Shop Operations Server | A, B, C, D, J, K, L, M, N, O, P |
| Database server | A, B, C, D, N, O, P |

TIP

In case of virtualization, monitoring should be performed for both VM and HW.

## Monitoring PharmaSuite Services

The following PharmaSuite-specific services can be monitored:

- PharmaSuite EBR server (page 72)
- PharmaSuite Transition server (page 77)
- PharmaSuite OE server (page 80)
- PharmaSuite TOM server (page 85)
- PharmaSuite ActiveMQ Broker (page 86)
- PharmaSuite AI server (page 86)

We recommend to set up a permanent monitoring solution which provides built-in support for monitoring Java processes and other system resources.

To monitor only the current values of the listed Java services, use JConsole (see [D5] (page 98)).

TIP

Since JConsole itself creates load and consumes memory on the related server, we recommend not to run it permanently but only when needed.

For convenience reason, the *wrapper.conf* files of the Java services already contain a section to configure remote monitoring with JMX including a suggestion for JMX ports. Please keep in mind that remote JMX requires security measures to ensure that unauthorized persons cannot control or monitor your application. For more information on how to secure remote JMX monitoring, see "Monitoring and Management Using JMX Technology - Java SE Monitoring and Management Guide" [D6] (page 98).

You can also monitor the heap space usage of the PharmaSuite services in the **About** dialog (page 65): the **Installation Details** dialog contains a section for each PharmaSuite service, including information about the current, maximum, and committed heap space. It also shows the heap memory usage as a percentage value of the maximum heap for convenience.

The following information applies to all PharmaSuite services when you monitor the available **heap space**: The server throws an OutOfMemory exception when the boundary value is reached. In most cases, the current processing fails and the server does not react anymore. We recommend to pay close attention to this parameter and consider to take preventive action.

## JMX MBeans Provided with PharmaSuite Services

PharmaSuite services provide some JMX MBeans (managed beans) that can be used to monitor the performance metrics and the current status of the services.

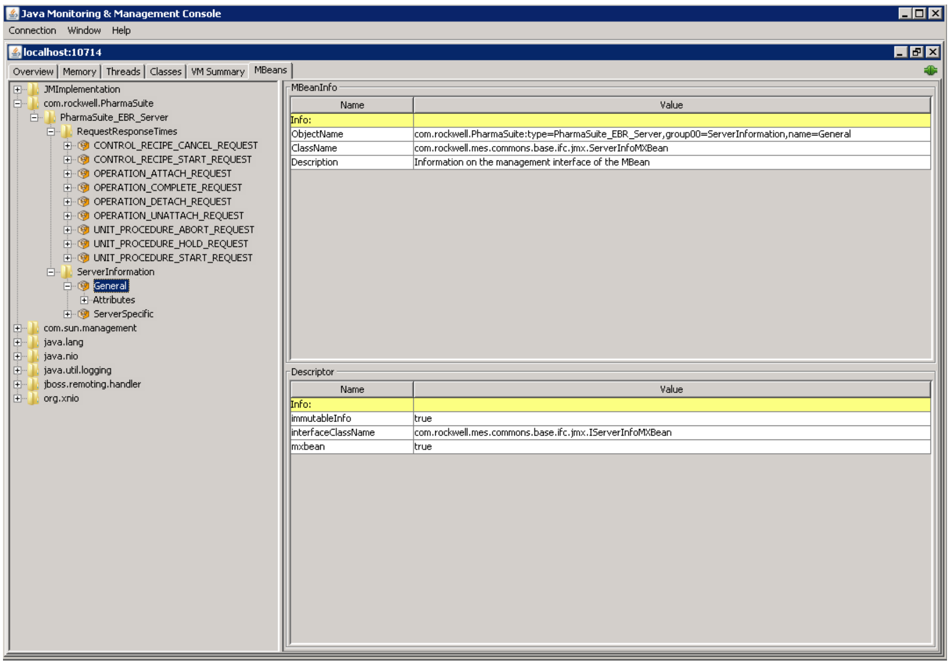
The beans that are common to all PharmaSuite services are described in the subsequent sections. For a description of beans that are specific for a PharmaSuite service, see the corresponding section of the PharmaSuite service.

The MBeans have a hierarchical structure determined by their names. The top node is always **com.rockwell.PharmaSuite** followed by a node with the name of the event sheet corresponding to the server. Underneath the nodes, the beans are grouped by their type.

Please use JConsole to review the available MBeans for setting up a monitoring solution.

**TIP**

When you set up a monitoring solution, please keep in mind that collecting the MBeans values in short intervals could affect the system performance. To avoid this, we recommend to set the collection interval to one minute or higher.



```
+ JMImplementation
- com.rockwell.PharmaSuite
   - PharmaSuite_EBR_Server
      - RequestResponseTimes
         + CONTROL_RECIPE_CANCEL_REQUEST
         + CONTROL_RECIPE_START_REQUEST
         + OPERATION_ATTACH_REQUEST
         + OPERATION_COMPLETE_REQUEST
         + OPERATION_DETACH_REQUEST
         + OPERATION_UNATTACH_REQUEST
         + UNIT_PROCEDURE_ABORT_REQUEST
         + UNIT_PROCEDURE_HOLD_REQUEST
         + UNIT_PROCEDURE_START_REQUEST
      - ServerInformation
         - General
            + Attributes
         + ServerSpecific
+ com.sun.management
+ java.lang
+ java.nio
+ java.util.logging
+ jboss.remoting.handler
+ org.xnio
```

MBeanInfo

| Name | Value |
|---|---|
| Info: | |
| ObjectName | com.rockwell.PharmaSuite:type=PharmaSuite_EBR_Server,group00=ServerInformation,name=General |
| ClassName | com.rockwell.mes.commons.base.ifc.jmx.ServerInfoMXBean |
| Description | Information on the management interface of the MBean |

Descriptor

| Name | Value |
|---|---|
| Info: | |
| immutableInfo | true |
| interfaceClassName | com.rockwell.mes.commons.base.ifc.jmx.IServerInfoMXBean |
| mxbean | true |

*Figure 5: Example of the EBR server MBeans in JConsole*

### General Server Info Bean

The bean contains some general information relevant to all PharmaSuite services.

### Request Response Time Beans

Each server provides information about the response times of all relevant ActiveMQ messages that are processed by the server. **Response time** is the time the server needs to process the message. One bean is created per message type, containing the following information (all times are in milliseconds):

- **AverageTimeRecent** is the average processing time for all messages received within the last 15 seconds. You can configure the time interval with the **MessageTypeTimeContainer.notOlderThanMillis** application configuration key. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).
  The number of messages used in the calculation is limited to 1000. You can change this setting with the **MessageTypeTimeContainer.recentArrayLimit** application configuration key. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).

- **AverageTimeSMA** is the simple moving average of the processing time. By default, the last 15 messages are used for the calculation. You can configure the number of messages to be used with the **MessageTypeTimeContainer.smaArrayLimit** application configuration key. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).
- **MinTime** is the minimum response time within the last N messages. The number of messages is the same as the one used for **AverageTimeSMA**.
- **MaxTime** is the maximum response time within the last N messages. The number of messages is the same as the one used for **AverageTimeSMA**.
- **TotalNumberOfRequests** is the total number of requests of this type received since the server was started.

The relevant message types are different for each PharmaSuite service. That is why the message types that will be provided via MBeans are configurable at a service level.

Three files are used for the configuration. The files are listed in preference order, i.e. if you change a property value in a file listed further down, it will overwrite the value from a file listed further up. The concept is similar to the one used for the PharmaSuite **log4j** configuration.

- A *jmx_messagetypes_ftps.properties* file is located in the *commons-messaging-ifc* JAR file. It contains the message types that are enabled for each PharmaSuite server by default.
- You can add a file named *jmx_messagetypes_custom.properties* in a custom JAR file of your choice.
- When a PharmaSuite server is started, an empty file called *jmx_messagetypes_custom_local.properties* is created in the *log* folder (unless it is already available).
  You can add new message types or remove message types that are enabled by adding entries to one of the two custom files. Each entry should have a key in the following format:
  **com.rockwell.PharmaSuite.JMX.<event sheet name>.<message type>**
  and a value of **true** or **false**.
  For example, if you do not want to monitor the CONTROL_RECIPE_START_REQUEST message type on the EBR server, add the following entry:
  ```
  com.rockwell.PharmaSuite.JMX.PharmaSuite_EBR_Server.CONTROL_
  RECIPE_START_REQUEST=false
  ```
  It overwrites the value in the *jmx_messagetypes_ftps.properties* file.
  You can edit the *jmx_messagetypes_custom_local.properties* file while the server is running. The changes will be detected within 60 seconds.

### Monitoring PharmaSuite EBR Server

The following items can be monitored.

| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>See information related to heap space in section "Monitoring PharmaSuite Services" (page 68). | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the SOS service.<br>Consider to adapt the maximum heap size. |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the SOS service. |
| **Perform sanity and communication checks.**<br>Sanity checks can be performed with the **Administrator Console** tool (**Check communication** action, see section "Checking the Communication Connection to the EBR and the OE server" (page 90)).<br>The check involves the messaging infrastructure including the PharmaSuite EBR server SOS service, which requests database server time by accessing the application server. | N/A | Check which component is affected (PharmaSuite ActiveMQ Broker SOS service, PharmaSuite EBR server SOS service, application server, database, or network infrastructure). |
| **Issues with individual orders/workflows** are reported to the affected Production Execution Clients or Production Management Clients. However, it can also happen that an issue prevents order steps from being available in the Production Execution Clients when they should be.<br>TIP<br>Procedures of orders and workflows can be unloaded from the EBR server and then loaded again with the **Administrator Console** tool (see section "Unloading and Reloading Procedures" (page 74)). | N/A | Check the error message and the log file of the EBR server. |

| Description | Boundary | Action |
|---|---|---|
| The **number of running unit procedures** can be monitored via JMX.<br>The number of unit procedures running on the EBR server affects the memory usage and the response time of the EBR server. Monitor this number if you expect that it will increase over time or if you notice a decrease in the performance (higher memory usage or response time). | > 6000 running unit procedures | Check if the performance is affected.<br>Double-check the reason for the high number of active orders and UPs.<br>Consider to restructure your recipes or unrelease orders that are not going to be processed immediately. If this is not applicable, consider using load balancing. |
| **Throughput of recipe execution** (**processing of unit procedures and operations per minute**) can be monitored by checking the number of UNIT_PROCEDURE_START_REQUEST and OPERATION_COMPLETE_REQUEST messages processed by the EBR server. The information is available via JMX. | > 60 unit procedures/minute<br>> 240 operations/minute | Check if the performance is affected.<br>Consider to restructure your recipes. If this is not applicable, consider using load balancing. |
| **Average response time of messages** can be monitored via JMX. If you notice an increase in the response time compared to the standard values, this could indicate an issue. | The response times increased more than three times than usual. | Check the health and load of your system. If the issue occurs only for one specific message type, check log files and contact the Rockwell Automation Customer support. |

### Monitoring All Nodes when Load Balancing Is Enabled

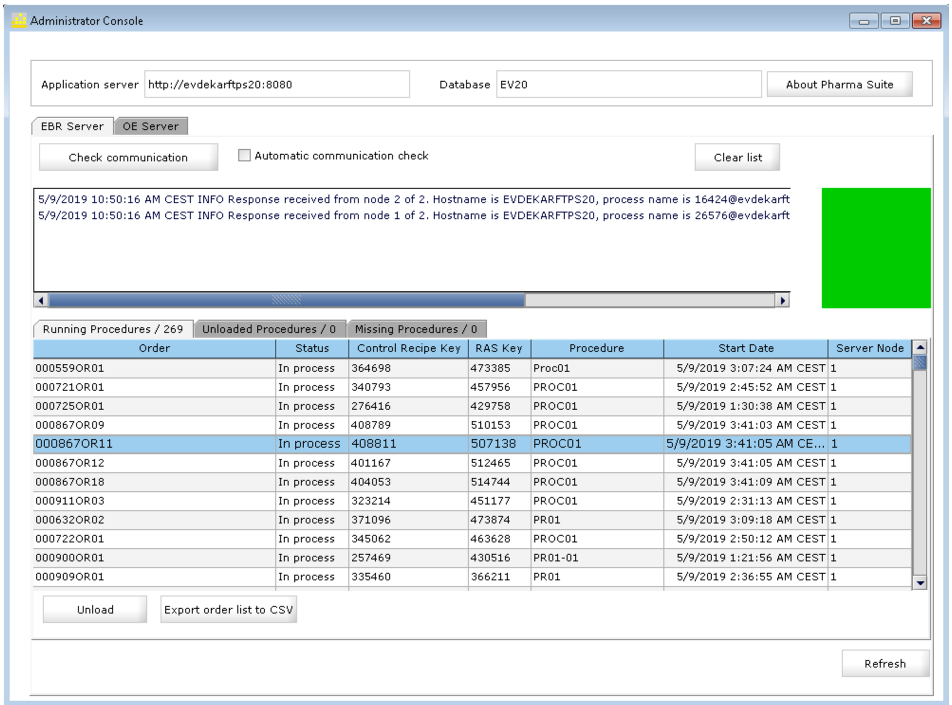
If you are experiencing performance issues on the EBR server caused by a high work load or if you plan to increase the production you may wish to consider enabling load balancing by setting up two or more EBR nodes. The active orders will be distributed between the nodes. For more information on how to setup multiple nodes, see "Technical Manual Installation" [A8] (page 97).

There are several specifics in monitoring multiple nodes:

- You must monitor the items listed above for every node.
- In the Administrator Console, you will be able to see on which server node each order is running. This can be used for troubleshooting purposes.
- When multiple nodes are configured, the server heartbeat check, the **Administrator Console**, and the **About** dialog are monitoring all of them. The specifics are described in the corresponding sections of this document.

### Unloading and Reloading Procedures

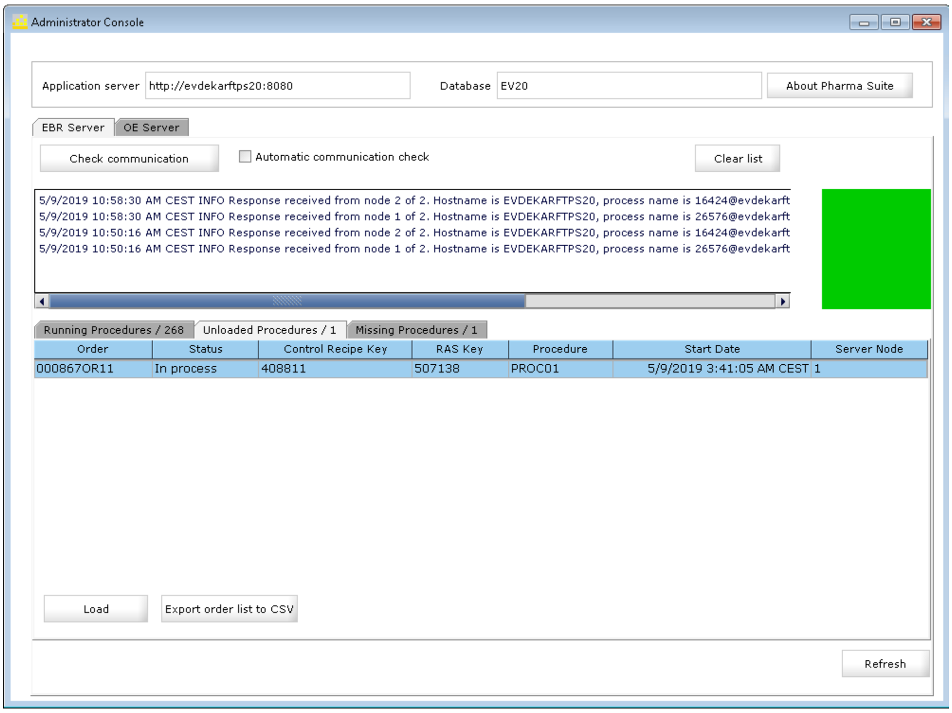
Procedures of orders and workflows running on the EBR server can be unloaded and reloaded later without shutting down the EBR server. This feature allows subject matter experts to correct corrupted processes (orders, workflows), e.g. by modifying entries in the database, while other processes can continue to run on the EBR server. PharmaSuite provides the **Administrator Console** tool for this purpose. The actions are controlled by one signature class (AC_EBR_ORDER_UNLOAD_AND_LOAD). After the electronic signature has been performed successfully, the system automatically records an "unload procedure"-specific exception for all unit procedures of the order and workflow or adds a "load procedure"-specific comment to the recorded exception. The risk level of the exception can be configured. For details, see chapter "Defining the Risk Level for Exceptions" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97).



```
5/9/2019 10:50:16 AM CEST INFO Response received from node 2 of 2. Hostname is EVDEKARFTPS20, process name is 16424@evdekarft
5/9/2019 10:50:16 AM CEST INFO Response received from node 1 of 2. Hostname is EVDEKARFTPS20, process name is 26576@evdekarft
```

| Order | Status | Control Recipe Key | RAS Key | Procedure | Start Date | Server Node |
|---|---|---|---|---|---|---|
| 000559OR01 | In process | 364698 | 473385 | Proc01 | 5/9/2019 3:07:24 AM CEST | 1 |
| 000721OR01 | In process | 340793 | 457956 | PROC01 | 5/9/2019 2:45:52 AM CEST | 1 |
| 000725OR01 | In process | 276416 | 429758 | PROC01 | 5/9/2019 1:30:38 AM CEST | 1 |
| 000867OR09 | In process | 408789 | 510153 | PROC01 | 5/9/2019 3:41:03 AM CEST | 1 |
| 000867OR11 | In process | 408811 | 507138 | PROC01 | 5/9/2019 3:41:05 AM CE... | 1 |
| 000867OR12 | In process | 401167 | 512465 | PROC01 | 5/9/2019 3:41:05 AM CEST | 1 |
| 000867OR18 | In process | 404053 | 514744 | PROC01 | 5/9/2019 3:41:09 AM CEST | 1 |
| 000911OR03 | In process | 323214 | 451177 | PROC01 | 5/9/2019 2:31:13 AM CEST | 1 |
| 000632OR02 | In process | 371096 | 473874 | PR01 | 5/9/2019 3:09:18 AM CEST | 1 |
| 000722OR01 | In process | 345062 | 463628 | PROC01 | 5/9/2019 2:50:12 AM CEST | 1 |
| 000900OR01 | In process | 257469 | 430516 | PR01-01 | 5/9/2019 1:21:56 AM CEST | 1 |
| 000909OR01 | In process | 335460 | 366211 | PR01 | 5/9/2019 2:36:55 AM CEST | 1 |

*Figure 6: Unload a procedure (order) that is running on the EBR server*



```
5/9/2019 10:58:30 AM CEST INFO Response received from node 2 of 2. Hostname is EVDEKARFTPS20, process name is 16424@evdekarft
5/9/2019 10:58:30 AM CEST INFO Response received from node 1 of 2. Hostname is EVDEKARFTPS20, process name is 26576@evdekarft
5/9/2019 10:50:16 AM CEST INFO Response received from node 2 of 2. Hostname is EVDEKARFTPS20, process name is 16424@evdekarft
5/9/2019 10:50:16 AM CEST INFO Response received from node 1 of 2. Hostname is EVDEKARFTPS20, process name is 26576@evdekarft
```

| Order | Status | Control Recipe Key | RAS Key | Procedure | Start Date | Server Node |
|---|---|---|---|---|---|---|
| 000867OR11 | In process | 408811 | 507138 | PROC01 | 5/9/2019 3:41:05 AM CEST | 1 |

*Figure 7: Load a procedure (order) that was unloaded from the EBR server*

To use the tool, run the **adminConsole** form to start the **Administrator Console** tool and navigate to the **EBR Server** tab.

To unload a running procedure, proceed as follows:

- Navigate to the **Running Procedures** tab. Select the procedure you wish to unload and click the **Unload** button. The **Administrator Console** tool checks the availability of the EBR server, requests an electronic signature, and instructs the EBR server to unload the procedure. If successful, the procedure is moved to the **Unloaded Procedures** tab. If not, the system displays an error message and the procedure remains in the **Running Procedures** tab.

**IMPORTANT - PERSISTENCE**

Unloading a procedure is not persistent. With a restart of the EBR server, all unloaded procedures, i.e. orders and workflows, are started normally.

This also means that there can be "unload procedure"-specific exceptions without a corresponding "load procedure"-specific comment in the batch or workflow report. Such a situation can occur when the EBR server is restarted while a procedure (order, workflow) is unloaded, since the procedure is loaded with the restart of the EBR server.

**IMPORTANT - ERROR CORRECTION**

How an order or workflow of the unloaded procedure can be corrected depends on the issue. Please always contact the support team and do not attempt to resolve an issue with solutions that were provided for other procedures unless the support team confirms the solution.

**IMPORTANT - ACTIONS IN THE PRODUCTION EXECUTION AND PRODUCTION MANAGEMENT CLIENTS**

The Cockpit of the Production Execution Client also lists the unit procedures and operations of an unloaded procedure (order, workflow) even though they cannot be started. If you try to start them, the system displays an appropriate message. Processing of already started operations is possible, since the Production Execution Client does not require to communicate with the EBR server during operation processing. In order to complete an operation and initializing the subsequent operation or completing the unit procedure, however, the Production Execution Client tries to contact the EBR server and thus requires the unit procedure to be reloaded to proceed. Please note that it can take up to ten minutes until the Production Execution Client is fully synchronized with the EBR server and regular processing can proceed after a unit procedure has been reloaded.

In the Production Management Client, for an unloaded procedure, i.e. its order, order steps, workflow, and workflow steps, actions are not disabled even though they cannot be performed. If you try to perform them, the system displays an appropriate message.

We highly recommend not to process unloaded procedures or to perform any actions on them.

To reload an unloaded procedure, proceed as follows:

- Navigate to the **Unloaded Procedures** tab. Select the procedure you wish to load and click the **Load** button. The **Administrator Console** tool checks the availability of the EBR server, requests an electronic signature, and instructs the EBR server to load the procedure. If successful, the procedure is moved to the **Running Procedures** tab. If not, the system displays an error message and the procedure remains in the **Unloaded Procedures** tab.

### EBR Server-specific MBean

There is an additional MBean for the EBR server containing information about the number of currently running orders and unit procedures.

## Monitoring PharmaSuite Transition Server

The following items can be monitored.

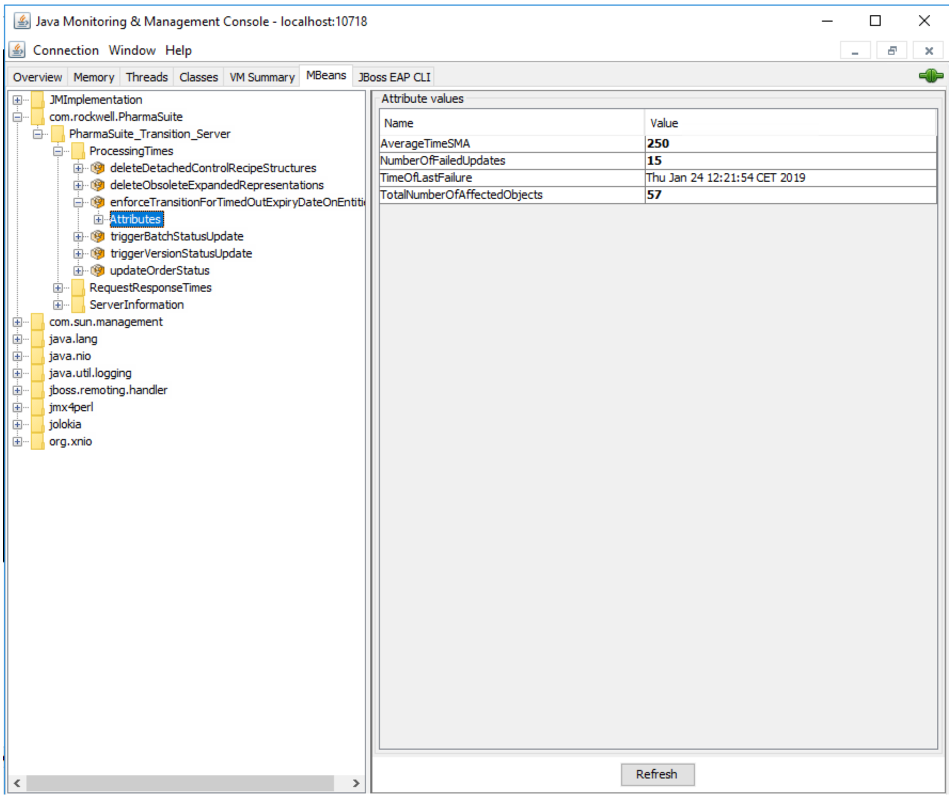
| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>See information related to heap space in section "Monitoring PharmaSuite Services" (page 68). | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the SOS service.<br>Consider to adapt the maximum heap size. |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the SOS service. |
| **Monitor** the **log files** of the **SOS service**.<br>For details, see **Code K** in "Monitoring FactoryTalk ProductionCentre Servers" (page 66).<br>It is important to review execution timeouts of the **delete detached control recipe structures** action. | N/A | Depends on message. |
| Check if there are procedures that were **not deleted successfully**. When the service fails to delete a procedure, it marks it as undeletable in the database.<br>The PharmaSuite **About** dialog (page 65) provides the number of procedures that are marked as undeletable, see section **Obsolete S88 structures** in the **Installation details** dialog. | Number of **undeletable procedures should be 0.** | Search the log files for the root cause.<br>If there are too many undeletable procedures, contact the Rockwell Automation Customer support. |

| Description | Boundary | Action |
|---|---|---|
| **Average response time of messages** can be monitored via JMX.<br>If you notice an increase in the response time compared to the standard values, this could indicate an issue. | The response times increased more than three times than usual. | Check the health and load of your system. If the issue occurs only for one specific message type, check log files and contact the Rockwell Automation Customer support. |
| **Average processing time and number of failed updates** can be monitored via JMX.<br>If you notice an increase of the processing time compared to the standard values, this could indicate an issue.<br>If the number of failed updates for a method is > 0 you should check the log file for the root cause. All errors are additionally logged in the *PharmaSuite_EBR_Server-ftps_transitionserver_error* log file in order to find them more easily. | The processing times increased more than three times than usual.<br>The number of failed updates for one of the methods is > 0. | For increased processing times, check the health and load of your system. For failed updates, check log files and contact the Rockwell Automation Customer support. |

### Accessing Details of the Transition Server and Its Events

The PharmaSuite Transition server uses MBeans (managed bean) to monitor the different actions performed on the server.

The *ProcessingTimes* MBean provides information about every action that is being performed by the Transition server.
The *RunningOperations* MBean provides details of all running operations (e.g. state, instance count).



Attribute values

| Name | Value |
|---|---|
| AverageTimeSMA | **250** |
| NumberOfFailedUpdates | **15** |
| TimeOfLastFailure | Thu Jan 24 12:21:54 CET 2019 |
| TotalNumberOfAffectedObjects | **57** |

*Figure 8: Example: attributes of an action performed by the Transition server displayed in JConsole as JMX client*

### Monitoring PharmaSuite OE Server

The following items can be monitored.

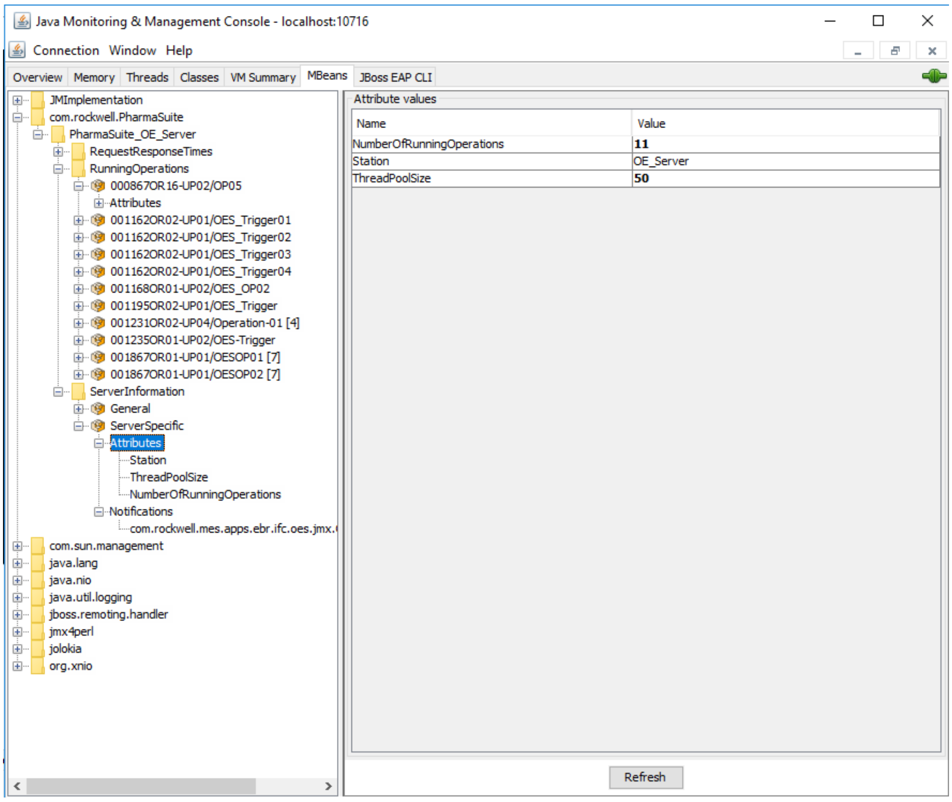
| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>See information related to heap space in section "Monitoring PharmaSuite Services" (page 68). | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the SOS service.<br>Consider to adapt the maximum heap size. |
| **Check** for the **number of running threads**. | > 300 | Verify the settings related to thread pooling (see section "Setting up Shop Operations Servers" in "Technical Manual Installation" [A8] (page 97). |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the SOS service. |
| **Perform sanity and communication checks**.<br>Sanity checks can be performed with the **Administrator Console** tool (**Check communication** action, see section "Checking the Communication Connection to the EBR and the OE server" (page 90)).<br>The check involves the messaging infrastructure including the PharmaSuite OE server SOS service, which requests database server time by accessing the application server. | N/A | Check which component is affected (PharmaSuite ActiveMQ Broker SOS service, PharmaSuite OE server SOS service, application server, database, or network infrastructure). |

| Description | Boundary | Action |
|---|---|---|
| **Issues on the OE server** are reported by error messages to affected Production Execution Clients.<br>If an issue is related to the execution of operations and phases, the system displays the error message at the running Production Execution Clients of the work centers of the unit procedure.<br>If an issue is not related to a specific work center, the system displays the error message at all running Production Execution Clients.<br>TIPS<br>■ The messages are also sent to a connected JMX client (e.g. JConsole) as notifications, if the user has subscribed.<br>■ Operations can be held and resumed with the **Administrator Console** tool (see section "Holding and Resuming Operations" (page 84)). | N/A | Check the error message and the log file of the OE server. |
| **Average response time of messages** can be monitored via JMX. If you notice an increase in the response time compared to the standard values, this could indicate an issue. | The response times increased more than three times than usual. | Check the health and load of your system. If the issue occurs only for one specific message type, check log files and contact the Rockwell Automation Customer support. |

### ACCESSING DETAILS OF THE OE SERVER AND ITS RUNTIME OPERATIONS

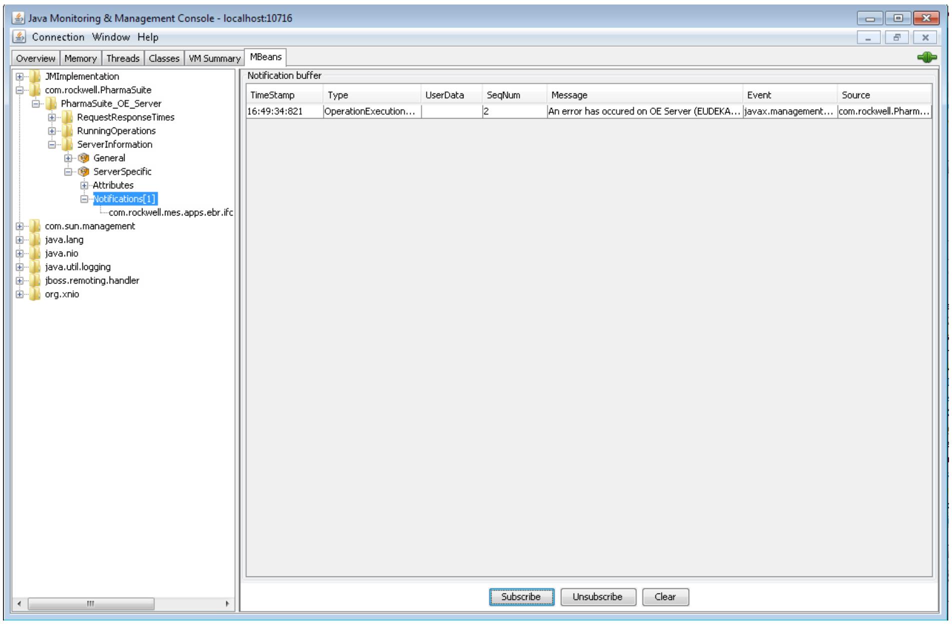
The PharmaSuite OE server uses MBeans (managed bean) to monitor OE server attributes and details of running operations.

The *ServerInformation/ServerSpecific* MBean provides some OE server attributes (e.g. station, device). If a user has subscribed for notifications, the error messages of the OE server are also available as notifications.
The *RunningOperations* MBean provides details of all running operations (e.g. state, instance count).


Java Monitoring & Management Console - localhost:10716
Connection Window Help
Overview Memory Threads Classes VM Summary MBeans JBoss EAP CLI
JMImplementation
com.rockwell.PharmaSuite
PharmaSuite_OE_Server
RequestResponseTimes
RunningOperations
000867OR16-UP02/OP05
Attributes
001162OR02-UP01/OES_Trigger01
001162OR02-UP01/OES_Trigger02
001162OR02-UP01/OES_Trigger03
001162OR02-UP01/OES_Trigger04
001168OR01-UP02/OES_OP02
001195OR02-UP01/OES_Trigger
001231OR02-UP04/Operation-01 [4]
001235OR01-UP02/OES-Trigger
001867OR01-UP01/OESOP01 [7]
001867OR01-UP01/OESOP02 [7]
ServerInformation
General
ServerSpecific
Attributes
Station
ThreadPoolSize
NumberOfRunningOperations
Notifications
com.rockwell.mes.apps.ebr.ifc.oes.jmx.
com.sun.management
java.lang
java.nio
java.util.logging
jboss.remoting.handler
jmx4perl
jolokia
org.xnio
Attribute values
Refresh

| Name | Value |
|---|---|
| NumberOfRunningOperations | **11** |
| Station | OE_Server |
| ThreadPoolSize | **50** |

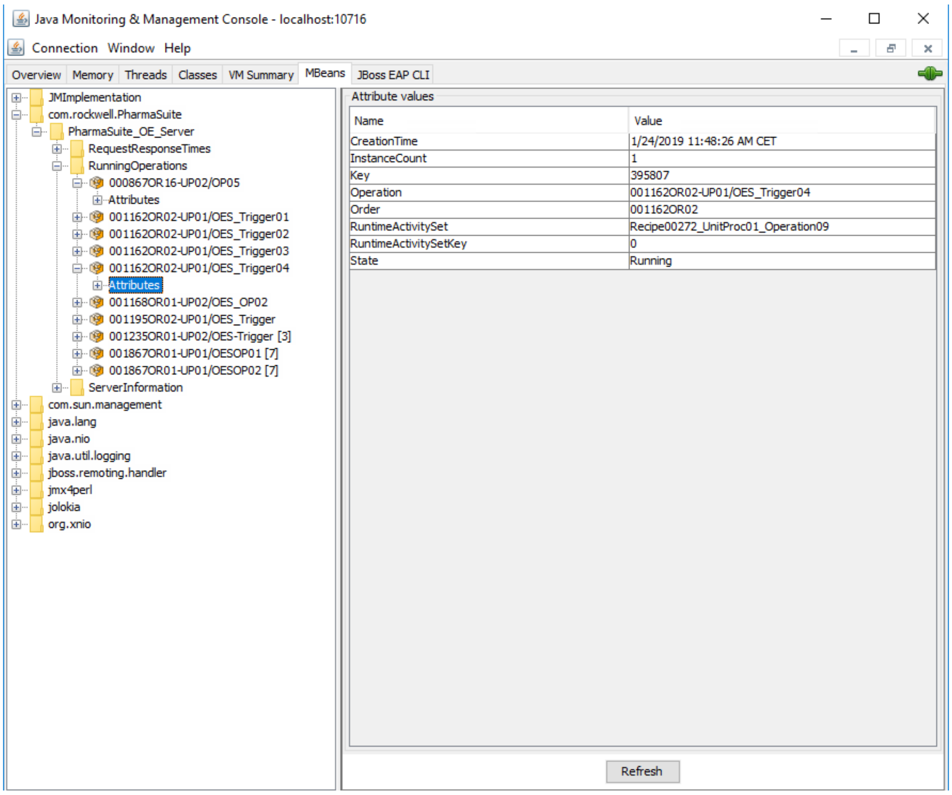
*Figure 9: Example: OE server attributes displayed in JConsole as JMX client*



Notification buffer

| TimeStamp | Type | UserData | SeqNum | Message | Event | Source |
|---|---|---|---|---|---|---|
| 16:49:34:821 | OperationExecution... |  | 2 | An error has occured on OE Server (EUDEKA... | javax.management... | com.rockwell.Pharm... |

*Figure 10: Example: OE server notifications displayed in JConsole as JMX client*



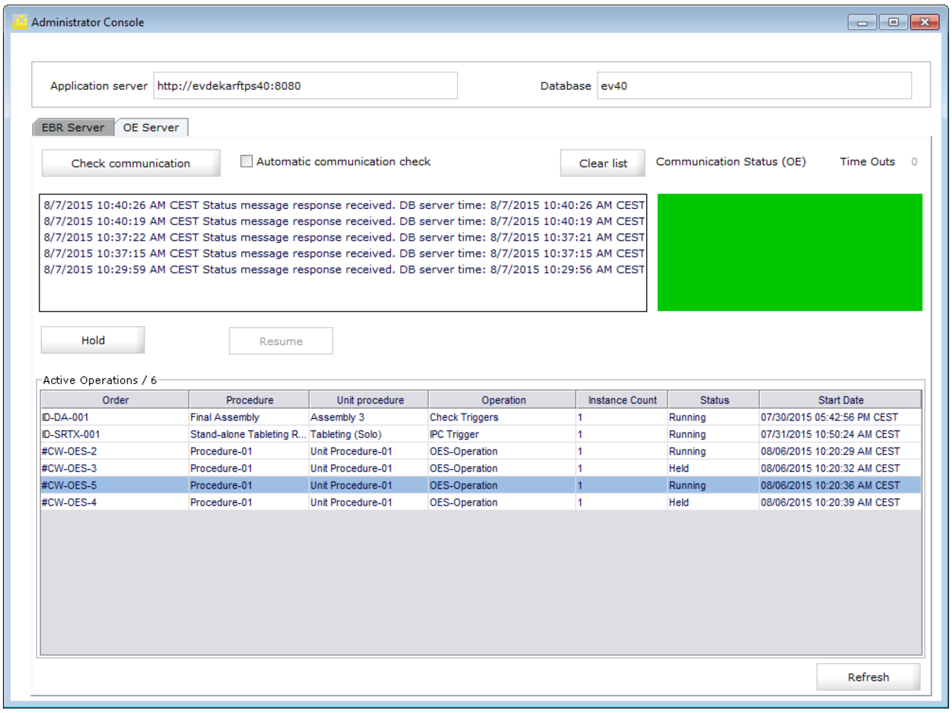
Attribute values

| Name | Value |
|---|---|
| CreationTime | 1/24/2019 11:48:26 AM CET |
| InstanceCount | 1 |
| Key | 395807 |
| Operation | 001162OR02-UP01/OES_Trigger04 |
| Order | 001162OR02 |
| RuntimeActivitySet | Recipe00272_UnitProc01_Operation09 |
| RuntimeActivitySetKey | 0 |
| State | Running |

*Figure 11: Example: running operation details displayed in JConsole as JMX client*

### Holding and Resuming Operations

Operations running on an OE server can be held and resumed later without a shutdown of the OE server. PharmaSuite provides the **Administrator Console** tool for this purpose. The actions are controlled by one signature class (AC_OES_OPERATION_HOLD_AND_RESUME). After the electronic signature has been performed successfully, the system automatically records a "hold operation"-specific exception for the operation or adds a "resume operation"-specific comment to the recorded exception. The risk level of the exception can be configured. For details, see chapter "Defining the Risk Level for Exceptions" in Volume 2 of the "Technical Manual Configuration and Extension" [A3] (page 97).



Application server http://evdekarftps40:8080

Database ev40

```
8/7/2015 10:40:26 AM CEST Status message response received. DB server time: 8/7/2015 10:40:26 AM CEST
8/7/2015 10:40:19 AM CEST Status message response received. DB server time: 8/7/2015 10:40:19 AM CEST
8/7/2015 10:37:22 AM CEST Status message response received. DB server time: 8/7/2015 10:37:21 AM CEST
8/7/2015 10:37:15 AM CEST Status message response received. DB server time: 8/7/2015 10:37:15 AM CEST
8/7/2015 10:29:59 AM CEST Status message response received. DB server time: 8/7/2015 10:29:56 AM CEST
```

| Order | Procedure | Unit procedure | Operation | Instance Count | Status | Start Date |
|---|---|---|---|---|---|---|
| ID-DA-001 | Final Assembly | Assembly 3 | Check Triggers | 1 | Running | 07/30/2015 05:42:56 PM CEST |
| ID-SRTX-001 | Stand-alone Tableting R... | Tableting (Solo) | IPC Trigger | 1 | Running | 07/31/2015 10:50:24 AM CEST |
| #CW-OES-2 | Procedure-01 | Unit Procedure-01 | OES-Operation | 1 | Running | 08/06/2015 10:20:29 AM CEST |
| #CW-OES-3 | Procedure-01 | Unit Procedure-01 | OES-Operation | 1 | Held | 08/06/2015 10:20:32 AM CEST |
| #CW-OES-5 | Procedure-01 | Unit Procedure-01 | OES-Operation | 1 | Running | 08/06/2015 10:20:36 AM CEST |
| #CW-OES-4 | Procedure-01 | Unit Procedure-01 | OES-Operation | 1 | Held | 08/06/2015 10:20:39 AM CEST |

*Figure 12: Hold and resume operations that are running on the OE server*

To use the tool, run the **adminConsole** form to start the **Administrator Console** tool and navigate to the **OE Server** tab.

To hold a running operation, proceed as follows:

- Select the operation you wish to hold and click the **Hold** button. The **Administrator Console** tool checks the availability of the OE server, requests an electronic signature, and instructs the OE server to hold the operation. If successful, the operation status is set **Held**. If not, the system displays an error message and the operation's status remains unchanged.

**IMPORTANT**
Manually held operations are intentionally **not** started when the OE server starts, they must be resumed manually.

To resume a held operation, proceed as follows:

- Click the **Resume** button. The **Administrator Console** tool checks the availability of the OE server, requests an electronic signature, and instructs the OE server to resume the operation. If successful, the operation status is set to **Running**. If not, the system displays an error message and the operation's status remains unchanged.

#### SERVER-SPECIFIC MBEAN

There is an additional MBean for the OE server containing information about the number of currently running operations.

### Monitoring PharmaSuite TOM Server

The following items can be monitored.

| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>See information related to heap space in section "Monitoring PharmaSuite Services" (page 68). | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the SOS service.<br>Consider to adapt the maximum heap size. |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the SOS service. |
| **Average response time of messages** can be monitored via JMX.<br>If you notice an increase in the response time compared to the standard values, this could indicate an issue. | The response times increased more than three times than usual. | Check the health and load of your system. If the issue occurs only for one specific message type, check log files and contact the Rockwell Automation Customer support. |

### Monitoring PharmaSuite ActiveMQ Broker

The following items can be monitored.

| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**. | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the service.<br>Consider to adapt the maximum heap size. |
| **Monitor queue sizes**. In the ActiveMQ WebConsole or via JMX, check the queue size of the DLQ and the number of pending messages in all queues. | The size of the **DLQ must be 0**.<br>The number of **pending messages must be 0 for all queues**. | Double-check the configuration of the ActiveMQ broker. |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the service. |

### Monitoring PharmaSuite AI Server

The following items can be monitored.

| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>See information related to heap space in section "Monitoring PharmaSuite Services" (page 68). | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the SOS service.<br>Consider to adapt the maximum heap size. |
| **Availability** check<br>The availability of this service is checked when the Production Execution Client is started and by cyclical server heartbeat checks within the Production Execution Client.<br>The PharmaSuite **About** dialog also provides details on the availability and response time of the service. | N/A | Restart the SOS service. |

| Description | Boundary | Action |
|---|---|---|
| Check if a tag can be read from Live Data via the AI server (see section "Checking the Communication Connection to the AI server" (page 87)). | Tag cannot be **read without error**. | Check the error message of the **Automation Integration Access Verification** tool.<br>Check the log file of AI server.<br>Resolve the problem.<br>It may be necessary to restart the AI server. |

#### AI Server-specific MBean

There is an additional MBean for the AI server containing the logical server name.

#### Checking the Communication Connection to the AI Server

The communication connection to the AI server can be checked with the **Automation Integration Access Verification** tool. For more information, see chapter "Setting up the Live Data Infrastructure for PharmaSuite" in "Technical Manual Installation" [A8] (page 97).

### Server Heartbeat Check

PharmaSuite runs a heartbeat check on its SOS servers to monitor their availability. If an issue is detected, the system displays a message to indicate this potential issue.

The following example shows how the heartbeat mechanism works for the EBR server:

1. The EBR server sends a heartbeat broadcast message to ActiveMQ broker every minute.
2. All Production Execution Clients are registered to listen for these broadcast messages.
3. The broadcast messages include the following relevant information:
   - Last heartbeat: Timestamp from the named SOS server machine, at the time the heartbeat broadcast message was sent by the SOS server.
   - Received by broker: Timestamp from the ActiveMQ broker machine, at the time when the heartbeat message was received and broadcasted by the ActiveMQ broker machine.

- Received by client: Timestamp on the Production Execution Client machine, at the time when the heartbeat broadcast message was received by the Production Execution Client.
- Host name: The name of the SOS server machine.
- Process name: The name associated with the SOS server process. It usually has the form *computerName@PID*, e.g. *server01@1234*.
- If multiple nodes are configured, the message contains the configuration of the node that is sending it, i.e. server node ID and total number of nodes.

4. Production Execution Clients receive broadcast message and check the different fields included in the message.
5. Production Execution Clients will show an error dialog if one of the following issues is detected:
   - "At least one required server seems to be unavailable."
     This message is displayed if a time difference between the EBR server machine and ActiveMQ or the Production Execution Client machines is detected or no heartbeat message is received at all. The error message will include the error type indicating what the exact root cause is:
     - STATUS_ERROR_TIME_MISMATCH if a heartbeat message was received, but the timestamp deviation between the EBR server and ActiveMQ or the Production Execution Client is bigger than three seconds to the current heartbeat message.
     - STATUS_ERROR if no heartbeat message was received in the last 120 seconds.

     Please follow up on the possible infrastructure issues that might be responsible for timestamp deviations:
     - The operating system time is not in sync among the different server and clients.
     - The SOS server cannot send a heartbeat message in time.
     - The heartbeat message is received or processed on the ActiveMQ broker with delay.
     - The heartbeat message is received or processed on the Production Execution Client with delay.
   - "At least one server is running as a new process."
     This message is displayed if two consecutive heartbeat messages from the same server contain a different host name or process ID. This can happen under the following circumstances:

    - The service representing the SOS server was restarted. If no restart was planned, this could indicate that something went wrong with the service (e.g. an out of memory error occurred).
    - If failover is configured, the failover mechanism is activated and the slave became master. This is not an issue per se, but if it happens too often, it could indicate a networking issue.
    - If multiple server nodes are configured and this error is displayed repeatedly, this could indicate that two nodes are configured with the same node ID. This is a serious issue, because it could lead to data inconsistency.

  If you see this error, check the log files of the corresponding SOS server and its node configuration.

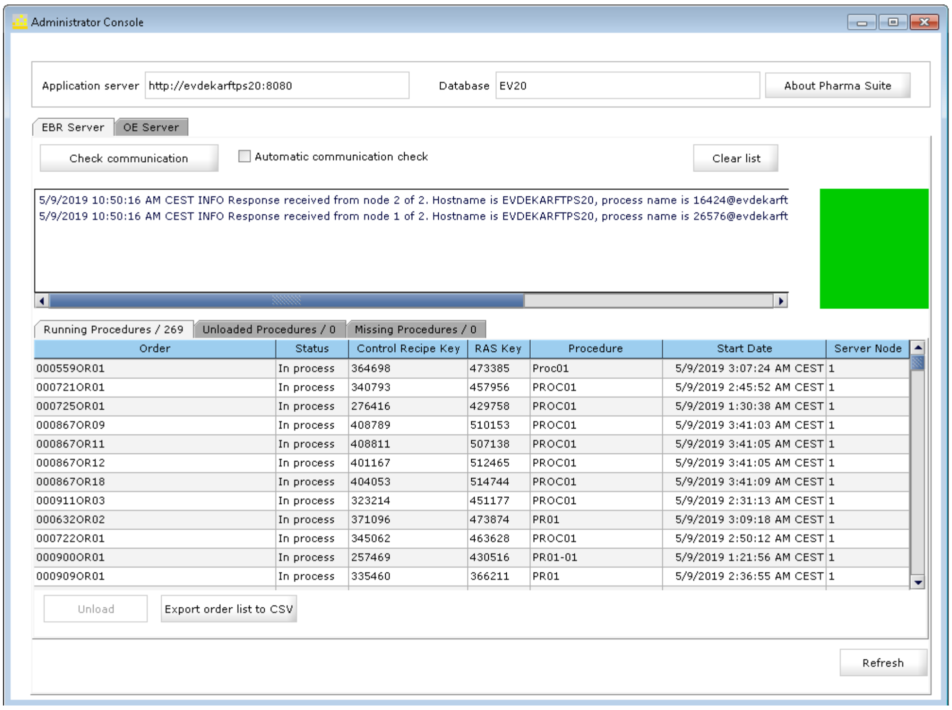
- "An inconsistency was detected in the load balancing configuration of at least one server."
  This message is displayed if multiple nodes are configured and two or more of them have an inconsistent configuration. For example, we could have the following two nodes:
    - EBR 1: node ID 1, number of nodes 1
    - EBR 2: node ID 2, number of nodes 2

  In this case, the number of nodes is different in both configurations, which could lead to data inconsistencies. If you see this error, check the configuration of the specified server nodes. It is recommended to stop production until the issue has been resolved.

## Checking the Communication Connection to the EBR and the OE Server

The execution of S88 recipes and workflows in PharmaSuite is realized as a Distributed Execution. Master recipes and master workflows consist of procedures, unit procedures, operations, and phases. A server component (EBR server) runs procedures and unit procedures, and the client runs operations and phases. Additionally, some specific operations run on an OE server.

PharmaSuite provides the **Administrator Console** tool to check the communication connection to the EBR server and the OE server either manually or automatically.



Application server http://evdekarftps20:8080

Database EV20

□ Automatic communication check

```
5/9/2019 10:50:16 AM CEST INFO Response received from node 2 of 2. Hostname is EVDEKARFTPS20, process name is 16424@evdekarft
5/9/2019 10:50:16 AM CEST INFO Response received from node 1 of 2. Hostname is EVDEKARFTPS20, process name is 26576@evdekarft
```

| Order | Status | Control Recipe Key | RAS Key | Procedure | Start Date | Server Node |
|---|---|---|---|---|---|---|
| 000559OR01 | In process | 364698 | 473385 | Proc01 | 5/9/2019 3:07:24 AM CEST | 1 |
| 000721OR01 | In process | 340793 | 457956 | PROC01 | 5/9/2019 2:45:52 AM CEST | 1 |
| 000725OR01 | In process | 276416 | 429758 | PROC01 | 5/9/2019 1:30:38 AM CEST | 1 |
| 000867OR09 | In process | 408789 | 510153 | PROC01 | 5/9/2019 3:41:03 AM CEST | 1 |
| 000867OR11 | In process | 408811 | 507138 | PROC01 | 5/9/2019 3:41:05 AM CEST | 1 |
| 000867OR12 | In process | 401167 | 512465 | PROC01 | 5/9/2019 3:41:05 AM CEST | 1 |
| 000867OR18 | In process | 404053 | 514744 | PROC01 | 5/9/2019 3:41:09 AM CEST | 1 |
| 000911OR03 | In process | 323214 | 451177 | PROC01 | 5/9/2019 2:31:13 AM CEST | 1 |
| 000632OR02 | In process | 371096 | 473874 | PR01 | 5/9/2019 3:09:18 AM CEST | 1 |
| 000722OR01 | In process | 345062 | 463628 | PROC01 | 5/9/2019 2:50:12 AM CEST | 1 |
| 000900OR01 | In process | 257469 | 430516 | PR01-01 | 5/9/2019 1:21:56 AM CEST | 1 |
| 000909OR01 | In process | 335460 | 366211 | PR01 | 5/9/2019 2:36:55 AM CEST | 1 |

*Figure 13: Check communication connection to the EBR server*

To use the tool, run the **adminConsole** form to start the **Administrator Console** tool and navigate to the **EBR Server** or **OE Server** tab, respectively.

The following functions are available:

- To perform a single manual check of the communication connection, proceed as follows:
    - Click the **Check communication** button. The **Administrator Console** tool sends a status request message to the EBR server or OE server. The reply is displayed in the list of status messages received from the EBR server or OE server and the **Communication Status (EBR/OE)** switches to green in case of a successful connection, otherwise the status turns red.
    - Click the **Clear list** button to clear the list of received status messages.
- To perform a repeated automatic check of the communication connection, proceed as follows:

- Select the **Automatic communication check** checkbox. Every two seconds, the **Administrator Console** tool sends a status request message to the EBR server or OE server. The reply is displayed in the list of status messages received from the EBR server or OE server and the **Communication Status (EBR/OE)** switches to green in case of a successful connection, otherwise the status turns red.
- Unselect the **Automatic communication check** checkbox to stop the automatic check.
- Click the **Clear list** button to clear the list of received status messages.

If multiple EBR nodes are configured, a status request message is sent to each node. Each status request will be sent twice to check if there are two different processes replying, which could indicate a misconfiguration. The **Communication Status** will switch to green only if all nodes are available and to red otherwise. The list of status messages will contain the replies from all nodes and the grid will contain the procedures running on all nodes. The **Server node** column contains the ID of the node where the corresponding procedure is running.

For the EBR server there are two additional tabs:

- The **Unloaded Procedures** tab is described in section "Unloading and Reloading Procedures" (page 74).
- The **Missing Procedures** tab contains all orders that are in the **Released** or the **In process** statuses but are not running on the EBR server. This could happen if load balancing is enabled and one of the EBR nodes is not running or if an order got corrupted and could not be loaded by the EBR server. This information can be used for troubleshooting.
- All three sub-tabs for the EBR server contain an **Export** button. It exports the contents of the corresponding grid as a .csv file.

## Debugging PharmaSuite Event Sheets

The **PharmaSuite_OE_Server**, **PharmaSuite_EBR_Server**, **PharmaSuite_AI_Server**, **PharmaSuite_TOM_Server**, and **PharmaSuite_Transition_Server** event sheets can be debugged with a debug configuration from a development environment (e.g. Eclipse). This is especially useful to debug phases that are running on the OE server.

The following prerequisites apply (based on Eclipse):

- An Eclipse workspace is available.
- All jar files located in the *SOS\lib* directory must be added to the class path.
- A FactoryTalk ProductionCentre server to test against has been set up.

To set up a debug configuration in Eclipse, proceed as follows:

1. Create the debug configuration.
    1. In Eclipse, from the **Run** menu, select the **Debug Configurations...** function.
    2. Right-click the **Java Application** node and select the **New** function.
    3. In the **Main** tab, set the **Main class** to **com.rockwell.eventmanager.EventManager**.
    4. Optional, in the **Arguments** tab, set the **VM arguments** as follows:
       ```
       -Xmx1024m
       -Dcom.datasweep.plantops.j2eevendor=JBoss
       -Duser.language=en -Duser.region=us
       -Djboss-ejb-client.reconnectOnAuthenticationFailures=false
       -Dsun.locale.formatasdefault=true  -
       Djava.net.preferIPv4Stack=true
       -Dorg.apache.activemq.SERIALIZABLE_PACKAGES=*
       -DuiDefaultButtonFollowFocus=false
       -DActivitySetContainerWithThreadPool.numberOfThreads=50
       ```
    5. If load balancing is enabled for the EBR server and you wish to debug one of the nodes, you must set the following **VM arguments** with the correct values:
       ```
       -DDLoadBalancingConfiguration.ServerNodeID=1
       -DDLoadBalancingConfiguration.NumberOfNodes=2
       ```

2. Create the event sheet configuration file.

   1. In your newly created debug configuration, click the **Debug** button in the lower right corner to start the debug configuration.
      Eclipse creates the XML configuration file and displays the path in the **Console** tab.
      Example: `ShopOperationsServer.xml at:`
      `C:\PATH_TO_YOUR_ECLIPSE_PROJECT\ShopOperationsServer.xml`

> **TIP**
> Usually, some error messages are displayed, but you can ignore them. The xml file is created nevertheless.

3. Adapt the event sheet configuration file.

   1. Open the XML file that was created by the first debug run.
      Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- The values listed below are for advanced users only. See
     the comments below for the functions of the values. All
these
     values have an upper limit and can not be less than 1. If
the
     value was set out of the range, it will be recognized as
the
     default value. Please do not change the other values in
this
     file directly. -->
<!-- function-thread-pool-size(int),
     default value:25,   max value:100  -->
<!-- main-processing-queue(int),
     default value:1000, max value:1000 -->
<!-- statistics-history-queue(int),
     default value:1000, max value:1000 -->
<!-- function-thread-processing-queue(int),
     default value:250,  max value:1000 -->
<shop-operations-server-configuration
  history-queue-enabled="true" log-file-count="4"
  ping-server-max-time="900" ping-server-duration="10"
  jetty-port="8084" statistics-enabled="true"
  main-processing-queue="1000" statistics-history-queue="1000"
  function-thread-processing-queue="250"
  function-thread-pool-size="25" log-file-size="2000000">
  <log-level>INFO</log-level>
  <http-uRL>http://localhost:8080</http-uRL>
  <username>admin</username><password>ozew#</password>
  <iiop-uRL>remote://localhost:8080</iiop-uRL>
  <log-folder>C:\Documents and Settings\User\logs</log-folder>
</shop-operations-server-configuration>
```

  2. Adapt the name and port of the server.
     Set up the event sheet to be executed by adding the `<event-sheet-name>` line.
     Always use `SERVER_NAME` instead of `localhost`.
     Example:

     ```
     <shop-operations-server-configuration
       history-queue-enabled="true" log-file-count="4"
       ping-server-max-time="900" ping-server-duration="10"
       jetty-port="8084" statistics-enabled="true"
       main-processing-queue="1000" statistics-history-queue="1000"
       function-thread-processing-queue="250"
       function-thread-pool-size="25" log-file-size="2000000">
       ...
       <http-uRL>http://SERVER_NAME:8080</http-uRL>
       ...
       <iiop-uRL>remote://SERVER_NAME:8080</iiop-uRL>
       ...
       <event-sheet-name>NAME_OF_THE_EVENT_SHEET</event-sheet-name>
     </shop-operations-server-configuration>
     ```

  3. Save your changes to the event sheet configuration file.

4. Run your debug configuration to debug the event sheet.

### Monitoring PharmaSuite Clients

The following items can be monitored.

| Description | Boundary | Action |
|---|---|---|
| **Monitor** the available **heap space**.<br>TIP<br>The PharmaSuite **About** dialog also provides details on the current values. | **> 80% of maximum heap space usage** | Provide a heap dump.<br>Restart the client.<br>Consider to adapt the maximum heap size. |
| **Monitor CPUs for sustained utilization over 80 %**. On multicore systems even one CPU having high load may indicate a potential problem or runaway process. | **> 80 %** for one CPU | Identify the process causing the issue. |
| **Monitor** for available **memory < 5 %** of total memory. Running out of memory may cause poor performance. | Available memory **< 5 %** | Identify the process causing the issue. |

| Description | Boundary | Action |
|---|---|---|
| **Monitor disk queue length for queues longer than 2.** If the queue, per physical disk, is longer than 2, this may indicate that your disks may be a performance bottleneck.<br>TIP<br>Disk queues can be monitored with Windows Performance Monitor. | Queues per physical disk longer than **2.** | Identify the process causing the issue. Check if the corresponding component has performance issues. If so, check the configuration of the component and/or consider to use hardware with better performance. |
| **Monitor** the **disk space.**<br>Running out of free disk space may cause data loss and computer hangs. | Free disk space **< 10 %** | Free disk space |

## Recovery Capabilities within PharmaSuite Clients

The **Production Execution Client** provides certain recovery capabilities to resolve issues during execution that are caused by a faulty or undefined recipe parameter setup (e.g. information flow expressions of a process parameter or transition conditions cannot be calculated by the system). These capabilities require specific access privileges.

- Abort phase execution
  The action column of the Navigator provides the capability to abort the execution of an active phase. When a phase is aborted, the system adds a "Phase execution aborted" exception to the aborted phase and enforces the completion of the phase. The forced completion bypasses all UI extensions (e.g. phase completion signature) and other phase-specific checks that may veto the completion.
  After the phase has been aborted, processing of the operation continues normally. That means the outgoing transition conditions of the aborted phase are evaluated and based on the evaluation successor phases will become active.

- Repair phase data
  The action column of the Navigator provides the capability to repair an active phase, i.e. to terminate the phase, create a new instance of the phase, and enable editing parameter data.

  - Repairing is not available for phases that are explicitly excluded from being repaired with the **PhaseRepairModeBlackList** configuration key. For details, see chapter "Configuration Keys of PharmaSuite" in Volume 4 of the "Technical Manual Configuration and Extension" [A4] (page 97).

- Repairing is only available as long as no exceptions have been recorded for a phase. This restriction does not include "phase repair mode"-specific exceptions.

When a phase is being repaired, the system adds a "Phase repair mode started" exception to the terminated phase.
In contrast to the abort action, after the phase has been terminated, the system does not continue to process its transition conditions. Instead, it creates a new instance of the terminated phase and puts the new instance into the processing context of the operation.
Before the new instance of the phase is started, the parameter data of the phase is editable in order to repair a misconfiguration. When the editing is confirmed, the system adds a "Phase parameters repaired" exception to the new instance of the phase that includes the original and new parameter data.

TIP
A repaired phase has a new instance count. Therefore recipes depending on instance count calculations may run into issues when they contain such phases.

The **Production Management Client** provides the capability to force an execution transition in case the transition is stalled during execution of a batch-specific order or a workflow. When an execution transition is forced, the system adds a "Force execution transition" exception to the related operation (force transition between phases), unit procedure (force transition between operations), or the unit procedure that was performed prior to the forced transition (force transition between unit procedures of an order).

# Reference Documents

The following documents are available from the Rockwell Automation Download Site.

| No. | Document Title | Part Number |
|---|---|---|
| A1 | PharmaSuite Implementation Guide PS Administration | PSAC-IN002B-EN-E |
| A2 | PharmaSuite Technical Manual Developing System Building Blocks | PSBB-PM010A-EN-E |
| A3 | PharmaSuite Technical Manual Configuration & Extension - Volume 2 | PSCEV2-GR010B-EN-E |
| A4 | PharmaSuite Technical Manual Configuration & Extension - Volume 4 | PSCEV4-GR010B-EN-E |
| A5 | FactoryTalk ProductionCentre Plant Operations Release 10.4 Server Installation Guide - JBoss Advanced | PCJBAD IN104A EN E |
| A6 | FactoryTalk ProductionCentre Plant Operations Release 10.4 Server Installation Guide - JBoss Stand-Alone | PCJBSA IN104A EN E |
| A7 | FactoryTalk ProductionCentre Administrator Release 10.4 User's Guide | PCADM-IN104A-EN-E |
| A8 | PharmaSuite Technical Manual Installation | PSES-IN010B-EN-E |
| A9 | PharmaSuite SDK Readme | N/A |

**TIP**

To access the Rockwell Automation Download Site, you need to acquire a user account from Rockwell Automation Sales or Support.

The following documents are distributed with the FactoryTalk ProductionCentre installation.

| No. | Document Title / Section |
|---|---|
| B1 | Process Designer Online Help |

**TIP**

To access the "Process Designer Online Help", use the following syntax: *http://<MES-PS-HOST>:8080/PlantOperations/docs/help/pd/index.htm*, where <MES-PS-HOST> is the name of your PharmaSuite server. To view the online help, the Apache Tomcat of the FactoryTalk ProductionCentre installation must be running.

The following documents are distributed with the PharmaSuite installation.

| No. | Document Title / Section |
|---|---|
| C1 | PharmaSuite-related Java Documentation: Interfaces of PharmaSuite |
| C2 | Data Manager User Documentation: Managing Equipment |

**TIP**

To access the "PharmaSuite-related Java Documentation", use the following syntax: *http://<MES-PS-HOST>:8080/PharmaSuite/javadoc/*, where <MES-PS-HOST> is the name of your PharmaSuite server.
To access the "Data Manager User Documentation", use the following syntax: *http://<MES-PS-HOST>:8080/PharmaSuite/documentationandhelp/index.htm*, where <MES-PS-HOST> is the name of your PharmaSuite server.

The following third-party documentation is available online as reference:

| No. | Document Title / Web Site |
|---|---|
| D1 | Mettler Toledo (http://www.mt.com) |
| D2 | Sartorius (http://www.sartorius.com) |
| D3 | Mettler Toledo IND780 Technical Manual: Communication, Configuration, Serial and PLC Interface |
| D4 | Troubleshooting Guide for HotSpot VM (http://docs.oracle.com/javase/7/docs/webnotes/tsg/TSG-VM/html/toc.html) |
| D5 | Using JConsole - Java SE Monitoring and Management Guide (http://docs.oracle.com/javase/7/docs/technotes/guides/management/jconsole.html) |
| D6 | Monitoring and Management Using JMX Technology - Java SE Monitoring and Management Guide (http://docs.oracle.com/javase/7/docs/technotes/guides/management/agent.html#gdenl) |
| D7 | Mettler Toledo IND890 Weighing Terminal User's Guide |

## Revision History

The following tables describe the history of this document.

Changes related to the document:

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Introduction" (page 1):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Extension and Naming Conventions" (page 3):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Integrating Custom Hardware" (page 7):

| Object | Description | Document |
|---|---|---|
| Integrating New Scales (page 7) | Reference to PharmaSuite SDK Readme added | 1.0 |

Changes related to "Best Practices for Managing User Accounts of a Client Operating System" (page 31):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Managing User and Access Rights" (page 33):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Exporting and Importing Master Recipes, Master Workflows, and Custom Building Blocks" (page 35):

| Object | Description | Document |
|---|---|---|
| All sections | Terminological clarification: the export/import feature refers to custom building blocks. | 1.0 |
| Customer-specific Parameter Attributes with References (page 49) | New section. | 1.0 |

Changes related to "Importing Equipment Classes and (Template) Entities from an External Source" (page 53):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Managing Device Identifiers" (page 55):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Localizing Each as Unit of Measure" (page 59):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Using Event Sheets in PharmaSuite" (page 61):

| Object | Description | Document |
|---|---|---|
| --- | --- | --- |

Changes related to "Monitoring PharmaSuite and Related Components" (page 65):

| Object | Description | Document |
|---|---|---|
| Unloading and Reloading Procedures (page 74) | IMPORTANT box: clarification added on processing of unloaded unit procedures in. | 1.0 |

# Index