SmartPlant Foundation 2008 (4.2) Configuration and Administration II

Course Guide Volume 1

INTERGRAPH

Version 4.2.1	February 2009	DSPF1-TP-100016A

SmartPlant Foundation 2008 (4.2) Configuration and Administration II

Course Guide Volume 1

February 2009 Version 4.2.1

DSPF1-TP-100016A

Copyright

Copyright © 2002 - 2009 Intergraph Corporation. All Rights Reserved.

Including software, file formats, and audiovisual displays; may be used pursuant to applicable software license agreement; contains confidential and proprietary information of Intergraph and/or third parties which is protected by copyright law, trade secret law, and international treaty, and may not be provided or otherwise made available without proper authorization.

Restricted Rights Legend

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) of the *Contractor Rights in Technical Data* clause at DFARS 252.227-7013, subparagraph (b) of the *Rights in Computer Software or Computer Software Documentation* clause at DFARS 252.227-7014, subparagraphs (b)(1) and (2) of the *License* clause at DFARS 252.227-7015, or subparagraphs (c) (1) and (2) of *Commercial Computer Software---Restricted Rights* at 48 CFR 52.227-19, as applicable.

Unpublished---rights reserved under the copyright laws of the United States.

Intergraph Corporation
Huntsville, Alabama 35894-0001

Warranties and Liabilities

All warranties given by Intergraph Corporation about equipment or software are set forth in your purchase contract, and nothing stated in, or implied by, this document or its contents shall be considered or deemed a modification or amendment of such warranties. Intergraph believes the information in this publication is accurate as of its publication date.

The information and the software discussed in this document are subject to change without notice and are subject to applicable technical product descriptions. Intergraph Corporation is not responsible for any error that may appear in this document.

The software discussed in this document is furnished under a license and may be used or copied only in accordance with the terms of this license.

No responsibility is assumed by Intergraph for the use or reliability of software on equipment that is not supplied by Intergraph or its affiliated companies. THE USER OF THE SOFTWARE IS EXPECTED TO MAKE THE FINAL EVALUATION AS TO THE USEFULNESS OF THE SOFTWARE IN HIS OWN ENVIRONMENT.

Trademarks

Intergraph, the Intergraph logo, SmartSketch, FrameWorks, SmartPlant, SmartPlant Foundation, SmartPlant P&ID, SmartPlant Instrumentation and INtools are registered trademarks of Intergraph Corporation. Microsoft and Windows are registered trademarks of Microsoft Corporation. Other brands and product names are trademarks of their respective owners.

This courseware was developed by PPM-PIM Training, Huntsville, Alabama.

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

Course Outline

Day 1

1. **SmartPlant Foundation 2008 Overview and Architecture**
   - Introduction to SPF Architecture
   - SmartPlant Foundation Authoring - Stand-Alone Mode
   - SmartPlant Foundation - Data Warehouse Mode
   - Combined Mode
   - SPF Containers

SPF 4.2 Configuration and Administration II (SPFA2)

- Data Model Layering
- Domains and Domain Groups
- SPF 2008 Database Tables
- Overview of Domains and Roles
- Data Warehouse Overview
- Managing Data Inconsistencies

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

2. Understanding the Loader Interface

- XML Load Format
- NLF Load File Format
- Loading Documents Using NLF
- Relationship Data
- Loading a Document with an Attached File
- Processing Load Files
- Reload Files
- Introduction to the Schema Import Wizard

SPF 4.2 Configuration and Administration II (SPFA2)

- Activity 1 – Loading SPF Users, Organizations and Role Assignments
- Activity 2 – Create Design Document Load File
- Activity 3 – Loading Schema Definitions with the Schema Import Wizard

3. Introduction to SPF Infrastructure

- Container Management
- System Administration User Preferences
- Expand Structure
- Subscriptions and Notifications
- Importing Schema Changes
- Creating Custom Notifications

SPF 4.2 Configuration and Administration II (SPFA2)

Day 2

- Activity 1 – Subscriptions and Notifications
- SmartPlant Foundation Scheduler
- Configuring SmartPlant Foundation E-mail
- Configuring E-mail Digests
- Custom Calendars
- Introduction to Vault Replication
- Activity 2 – SPF Scheduler and Calendars

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

4. Overview of Forms

- Generating a Form
- Form Editing with Form Builder
- Modifying Data Form Sections
- Modifying Display Items
- Activity 1 – Creating Forms
- Creating Forms Manually
- Copying Forms
- Activity 2 – Manually Creating Forms

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

5. The SPF Administrative GUI

- Introduction to API's and Methods
- Client API's
- Conditions
- Configuring New Methods
- Manage Related Items
- View Related Items
- Relating New Classes to a Domain

Activity 1 – Configuring New Methods

SPF 4.2 Configuration and Administration II (SPFA2)

- Creating and Managing Menus and Menu Items
- Managing Sub Menus
- Creating Toolbars
- Quick Find Menu Items
- Activity 2 – Creating Menus and Menu Items
- Tree View Display Column Set
- Enumeration Lists and Hierarchies
- Activity 3 – Creating Column Sets

SPF 4.2 Configuration and Administration II (SPFA2)

Day 3

6. Introduction to Classified Objects

- Modeling the Classification
- Loading and Viewing Classified Definitions
- Viewing the Classification Tree
- Creating a New Form for Classification
- Creating Methods for Classified Objects
- Creating Menu Items for Classified Objects
- Creating a Drill Down Tree
- Configuring the Engineering Numbering System

SPF 4.2 Configuration and Administration II (SPFA2)

- Testing Homogeneous Classifications
- Configuring and Testing Heterogeneous Classifications
- Activity - Creating Classified Objects

7. Introduction to SmartPlant Foundation Workflow

(with an interactive activity during the chapter)

- Creating Workflows
- Distribution Matrix Overview
- Building a Workflow
- Viewing Workflows
- Attaching Workflows

SPF 4.2 Configuration and Administration II (SPFA2)

- Workflow Options
- Using Workflows
- Submitting a Document to the Workflow
- Workflow Signoffs
- Workflow History

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

8. Introduction to Transmittals

- Transmittal Project Roles
- Transmittal Structure
- Transmittal Administrative Setup
- Transmittal Users and Roles
- Modifying Transmittal Workflows
- Distribution Matrix Overview

Activity 1 – Transmittal Administrative Setup

SPF 4.2 Configuration and Administration II (SPFA2)

Day 4

- Creating an Internal Transmittal
- Creating a Transmittal Section
- Expanding the Transmittal Structure
- Creating an External Transmittal
- Saving a Transmittal Template
- Create a Transmittal for a Selected Set of Documents
- Creating a Transmittal from a Template

Activity 2 – Creating Transmittals

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

9. Using and Managing Transmittals

- Searching for Transmittals
- Processing a Transmittal
- Preparing a Transmittal
- Approve Transmittal
- Exporting a Transmittal Structure
- Issuing a Transmittal and Recipient Response
- Completing a Transmittal
- Transmittal Manipulations
- Activity – Using and Managing Transmittals

SPF 4.2 Configuration and Administration II (SPFA2)

10. Creating and Using Datasheets

- Working with Datasheets
- The SmartPlant Schema
- Creating a Custom Interfaces and Properties
- Creating a Datasheet Graph Definition
- Creating a Datasheet View Definition
- CMF File Manager
- Importing Datasheet Schema Additions
- Loading Additional Relationships

Activity 1 – Schema Modeling to Support Datasheets

SPF 4.2 Configuration and Administration II (SPFA2)

- Creating New Datasheet Templates
- Creating a Datasheet Definition
- Using the Datasheet Definer
- Mapping a String Property Value
- Mapping a UOM Property Value
- Saving the Datasheet Mapping

Activity 2 – Creating Datasheet Template Documents, Datasheet Definitions and Using Datasheet Definer

SPF 4.2 Configuration and Administration II (SPFA2)

- Registering an Authoring Tool
- Retrieving a Published Instrument
- Defining Process Cases
- Testing Datasheet Functionality
- Using Datasheets Offline
- Loading Offline Datasheet Updates

Activity 3 – Testing Custom Datasheets

SmartPlant® Foundation

SPF 4.2 Configuration and Administration II (SPFA2)

11. Adhoc Reporting

- Creating a Report Graph Definition
- Creating a Report View Definition
- Custom Line List Adhoc Report Contents
- Creating a Graph Definition for the Line List Report
- Creating a View Definition for the Line List Report
- Importing Report Schema Additions
- Generating a Report
- Additional Adhoc Report Options
- Generating a Line List Adhoc Report

Activity – Creating and Running AdhocReports

SPF 4.2 Configuration and Administration II (SPFA2)

Day 5

12. Introduction to the Excel Import

- Excel Import Setup
- Preparing the Import File
- Importing the Objects

13. Creating Plant Hierarchies

- Creating the Custom Hierarchy in the Schema Editor
- Loading the New Hierarchy into the Database
- Create Admin Objects for the New Hierarchy

SPF 4.2 Configuration and Administration II (SPFA2)

- Update the PBS Map Files
- Test the Custom PBS Hierarchy
- Activity – Creating a Custom Plant Hierarchy

Wrap up and Dismissal

Table of Contents

1. SmartPlant Foundation 2008 Overview and Architecture __ 1-3

1.1 Introduction to SPF Architecture _____ 1-4
1.1.1 SmartPlant Foundation Authoring - Stand-Alone Mode _____ 1-12
1.1.2 SmartPlant Foundation - Data Warehouse Mode _____ 1-14
1.1.3 Combined Mode _____ 1-15

1.2 SPF Containers _____ 1-16

1.3 Data Model Layering _____ 1-18

1.4 Domains and Domain Groups _____ 1-24
1.4.1 SPF 2008 Database Tables _____ 1-25

1.5 Overview of Domains and Roles _____ 1-30

1.6 Data Warehouse Overview _____ 1-35
1.6.1 Managing Data Inconsistencies _____ 1-41

2. Understanding the Loader Interface _____ 2-3

2.1 XML Load Format _____ 2-7
2.1.1 XML Loader Instructions _____ 2-9

2.2 NLF Load File Format _____ 2-12
2.2.1 NLF Instructions _____ 2-16
2.2.2 Field Types for NLF _____ 2-19

2.3 Loading Documents Using NLF _____ 2-21
2.3.1 Relationship Data _____ 2-23
2.3.2 Loading a Document with an Attached File _____ 2-26

2.4 Processing Load Files _____ 2-29
2.4.1 Additional SPF Loader Features _____ 2-37

2.5 Reload Files _____ 2-44

2.6 Introduction to the Schema Import Wizard _____ 2-46

2.7 Activity 1 – Loading SPF Users, Organizations and Role Assignments _____ 2-53

2.8 Activity 2 – Create Design Document Load File _____ 2-55

2.9 Activity 3 – Loading Schema Definitions with the Schema Import Wizard _____ 2-57

3. Introduction to SPF Infrastructure _____ 3-3

3.1 Container Management _____ 3-4

3.2 System Administration User Preferences _____ 3-17
3.2.1 Display Objects Using Internal Names _____ 3-18

3.3 Expand Structure _____ 3-24

3.4 Subscriptions and Notifications _____ 3-27
3.4.1 Enabling Notification List Access _____ 3-29
3.4.2 Setting Up Subscriptions _____ 3-33

3.5 Importing Schema Changes 3-43
3.5.1 Creating Log Subscriptions 3-48

3.6 Creating Custom Notifications 3-52
3.6.1 Viewing Custom Notifications 3-59

3.7 Activity 1 – Subscriptions and Notifications 3-61

3.8 SmartPlant Foundation Scheduler 3-63

3.9 Configuring SmartPlant Foundation E-mail 3-67
3.9.1 Testing the E-mail Configuration 3-69

3.10 Configuring E-mail Digests 3-73
3.10.1 Email Digest Setup 3-75

3.11 Custom Calendars 3-77

3.12 Introduction to Vault Replication 3-83
3.12.1 Setting up a standalone File Server 3-84
3.12.2 Setting up Vaults for Replication 3-85
3.12.3 Replicated Vault Viewing 3-91
3.12.4 File Replication Mechanisms **Error! Bookmark not defined.**
3.12.5 Using Robocopy for SPF Replication **Error! Bookmark not defined.**
3.12.6 Constant Synchronization **Error! Bookmark not defined.**
3.12.7 Periodic Synchronization **Error! Bookmark not defined.**
3.12.8 Recovery options **Error! Bookmark not defined.**
3.12.9 Installation as a Service **Error! Bookmark not defined.**

3.13 Activity 2 – SPF Scheduler and Calendars 3-101

4. Overview of Forms 4-3

4.1 Generating a Form 4-5

4.2 Form Editing with Form Builder 4-8
4.2.1 Modifying Data Form Sections 4-13
4.2.2 Modifying Display Items 4-17

4.3 Activity 1 – Creating Forms 4-26

4.4 Creating Forms Manually 4-27
4.4.1 Creating Sections Manually 4-29
4.4.2 Creating Display Items Manually 4-34
4.4.3 Copying Forms 4-48

4.5 Activity 2 – Manually Creating Forms 4-51

5. The SPF Administrative GUI 5-3

5.1 Introduction to API's and Methods 5-6
5.1.1 Client API's 5-7
5.1.2 Conditions 5-9

5.2 Configuring New Methods 5-17
5.2.1 Common Method APIs 5-23
5.2.2 Manage Related Items 5-25
5.2.3 View Related Items 5-32

5.2.4 Relating New Classes to a Domain 5-34

5.3 Activity 1 – Configuring New Methods 5-39

5.4 Creating and Managing Menus and Menu Items 5-41

5.4.1 Managing Sub Menus 5-46

5.5 Creating Toolbars 5-52

5.6 Quick Find Menu Items 5-54

5.7 Activity 2 – Creating Menus and Menu Items 5-57

5.8 Tree View Display Column Set 5-59

5.9 Configuring Additional Methods and Menus 5-71

5.10 Enumeration Lists and Hierarchies 5-83

5.11 Activity 3 – Creating Column Sets 5-87

6. Introduction to Classified Objects 6-3

6.1 Modeling the Classification 6-6

6.1.1 Importing the Model Additions 6-22

6.1.2 Modeling a Document Classification Object 6-26

6.1.3 Loading and Viewing Classified Definitions 6-33

6.2 Viewing the Classification Tree 6-41

6.3 Creating a New Form for Classification 6-47

6.3.1 Loading the Form Relationship 6-59

6.4 Creating Methods for Classified Objects 6-60

6.5 Creating Menu Items for Classified Objects 6-66

6.6 Creating a Drill Down Tree 6-70

6.6.1 Adding Method Access Rights 6-72

6.7 Configuring the Engineering Numbering System 6-78

6.7.1 Relating the ENS to the Classification 6-83

6.8 Testing Homogeneous Classifications 6-86

6.9 Configuring and Testing Heterogeneous Classifications 6-91

6.10 Activity – Creating Classified Objects 6-97

7. Introduction to SmartPlant Foundation Workflow 7-3

7.1 Creating Workflows 7-5

7.1.1 Step Types 7-6

7.1.2 Step Classes 7-9

7.1.3 Create a Step Class 7-12

7.1.4 Step Definitions 7-17

7.1.5 Create a Step Definition 7-21

7.1.6 Using Reasons for Receipt 7-23

7.1.7 Create a Reason For Receipt 7-25

7.1.8 Workflow Check Lists 7-28
7.1.9 Workflow Statuses 7-33

7.2 Distribution Matrix Overview 7-36
7.2.1 Creating a Document Matrix 7-38

7.3 Building a Workflow 7-45

7.4 Viewing Workflows 7-63

7.5 Attaching Workflows 7-66
7.5.1 Workflow Access Configuration 7-70

7.6 Workflow Options 7-76

7.7 Using Workflows 7-80
7.7.1 Setting the Active Scope 7-81

7.8 Submitting a Document to the Workflow 7-83
7.8.1 Attaching an Object to a Workflow 7-87
7.8.2 Viewing an Active Workflow 7-89
7.8.3 Workflow Signoffs 7-94

7.9 Workflow History 7-110

CHAPTER 1

SmartPlant Foundation Overview and Architecture

1. SmartPlant Foundation 2008 Overview and Architecture

This chapter is intended to provide an overview of the 2008 version of SmartPlant Foundation. The first part of the chapter will cover some of the changes to the architecture of SPF 2008. The remainder of the chapter will introduce some of the features included in this software.

1.1 Introduction to SPF Architecture

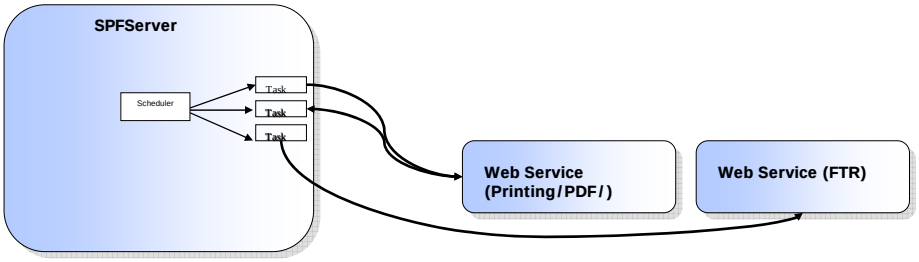
As on overview of SPF 2008, the architecture contains the following peripheral products:

- PDF generation uses the same procedure as SPF 2007.
- The version of SmartPlant Markup and Markup Plus used in the class VM session is 03.08.04.10 SP4.

In SPF 2008, the scheduler is no longer a separate client. It now runs as a thread in the server.

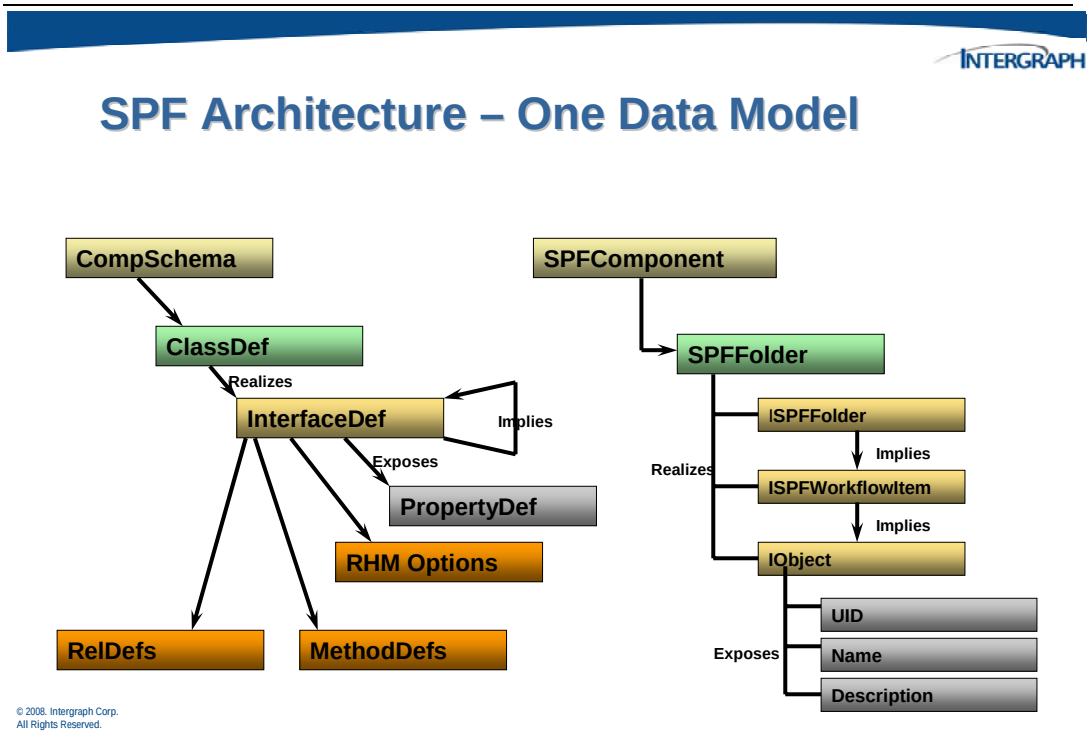
Introduction to SPF Architecture

SPF 4.2 Scheduler



© 2008. Intergraph Corp.
All Rights Reserved.

Unlike the previous versions, which featured a Business Object (BO) model and a Schema Object (SO) model, SPF 2008 has a single data model.

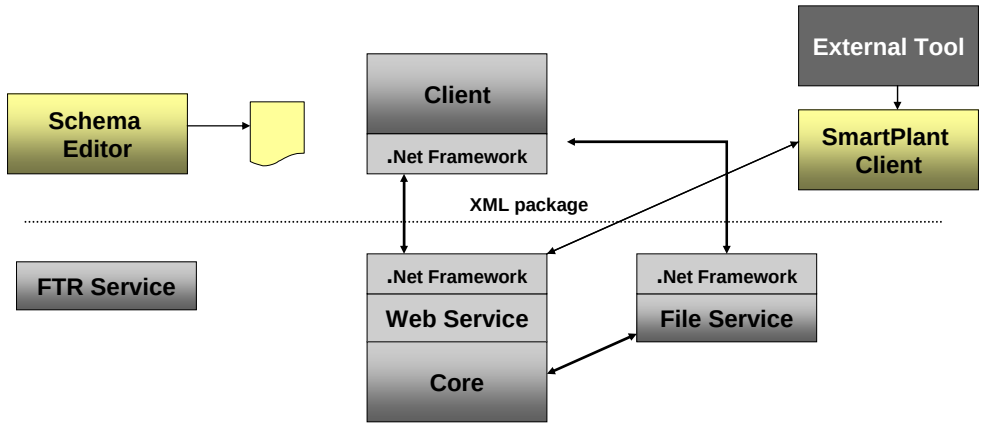


The interface definition has relationships to properties, dynamic menus (RHM menus), methods and relationship definitions.

An example is shown on the right side of the diagram above with the **SPFFolder** class.

In SPF 2008, all modeling is done in the Schema Editor, which writes the results to an XML file. The xml model files are loaded to the SPF database using either the ***File*** > ***Loader*** or the ***Administration*** > ***Schema Import Wizard*** commands in the Desktop Client. All communication with the SPF server is routed through *.Net Framework* layer of the Desktop Client. Engineering authoring tools use the **SmartPlant Client** to communicate with the *.Net Framework* layer at the SPF server level.

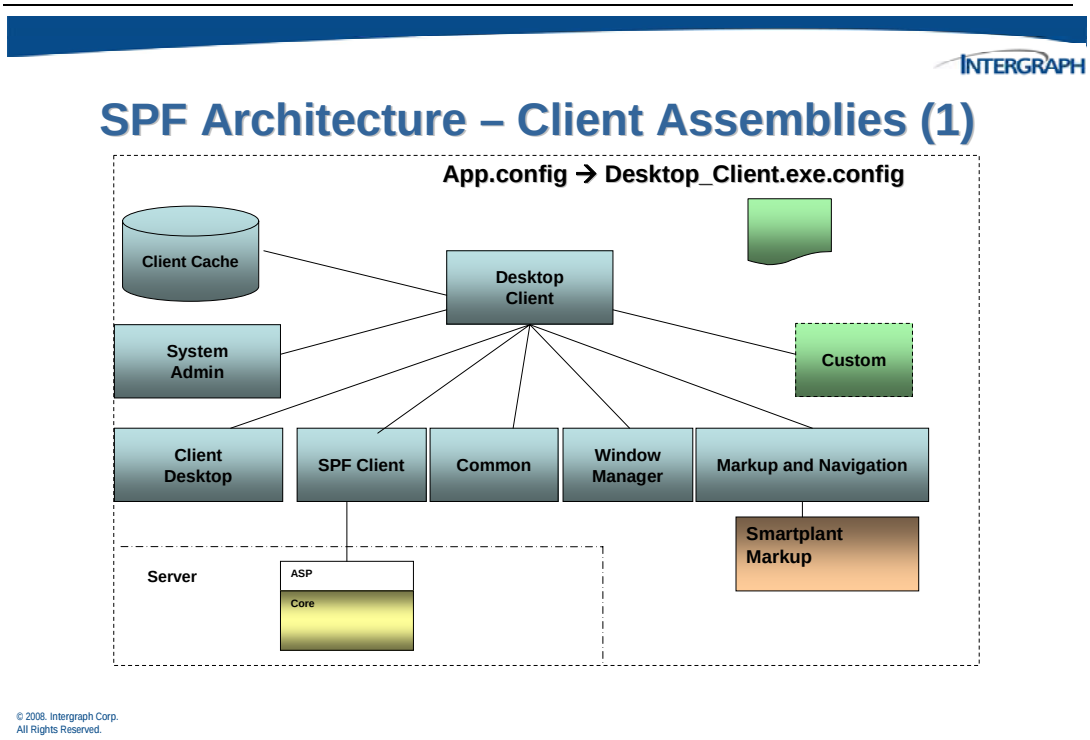
SPF Architecture – Key Assemblies



© 2008. Intergraph Corp.
All Rights Reserved.

The server architecture contains the *.Net Framework*, *Web Service* and *Core* layers. The architecture for a File server consists of the *.Net Framework* and *File Service* layers. When a client communicates with a file server, it is done through XML packages.

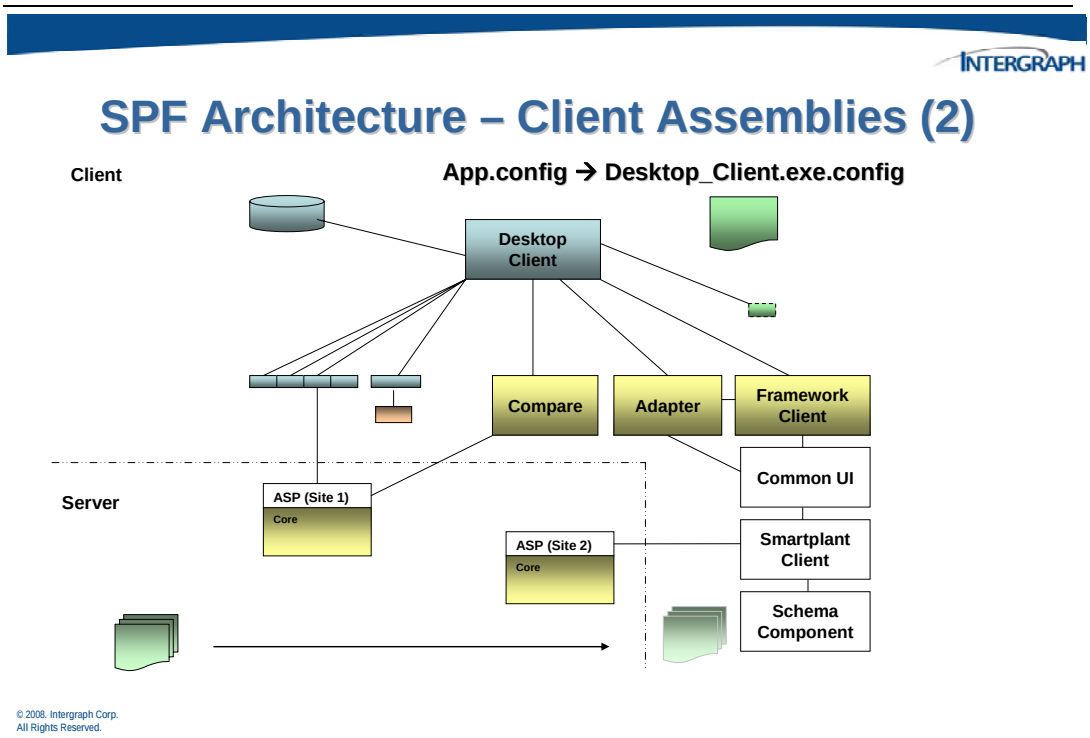
When a client starts, a set of assemblies get loaded. These assemblies are *Desktop Client*, *SPF Client*, *Common*, *Window Manager* and *Markup and Navigation*.



The *Common* assembly contains a common set of things such as exception handling and tracing.

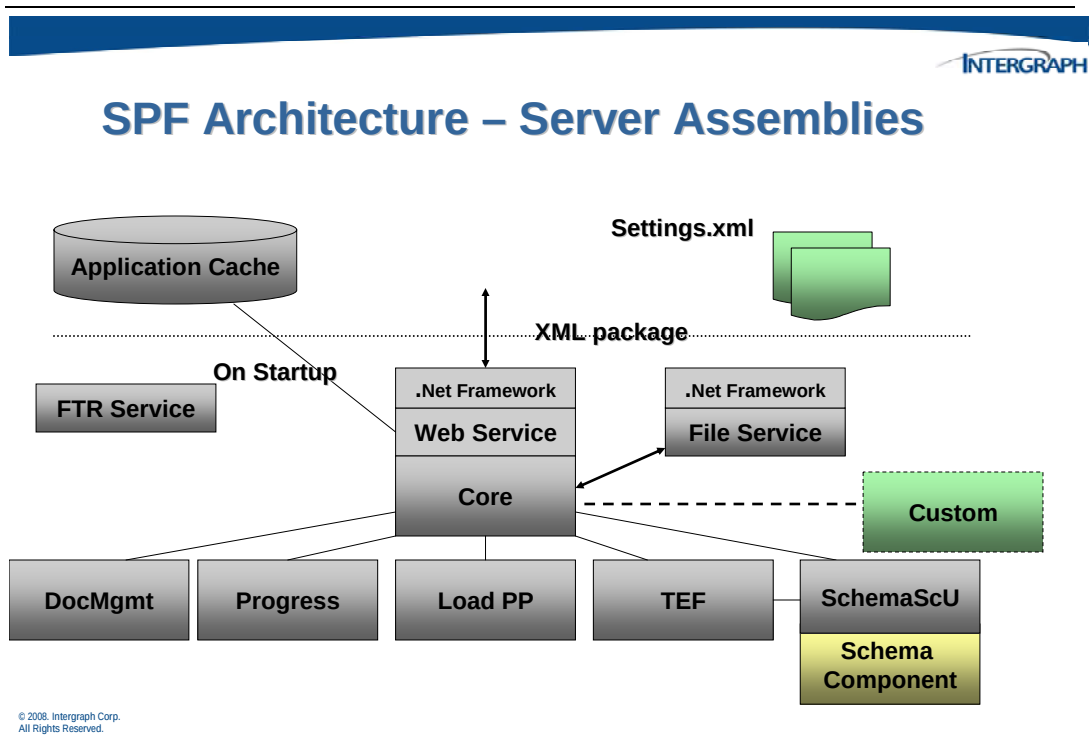
The *App.config* file becomes the **Desktop_Client.exe.config** file (on the client) when it is compiled. If a custom assembly needs to be loaded at client start, then it has to first be added to the Desktop_Client.exe.config.

The SPF 2008 client is split for authoring with or without integration.



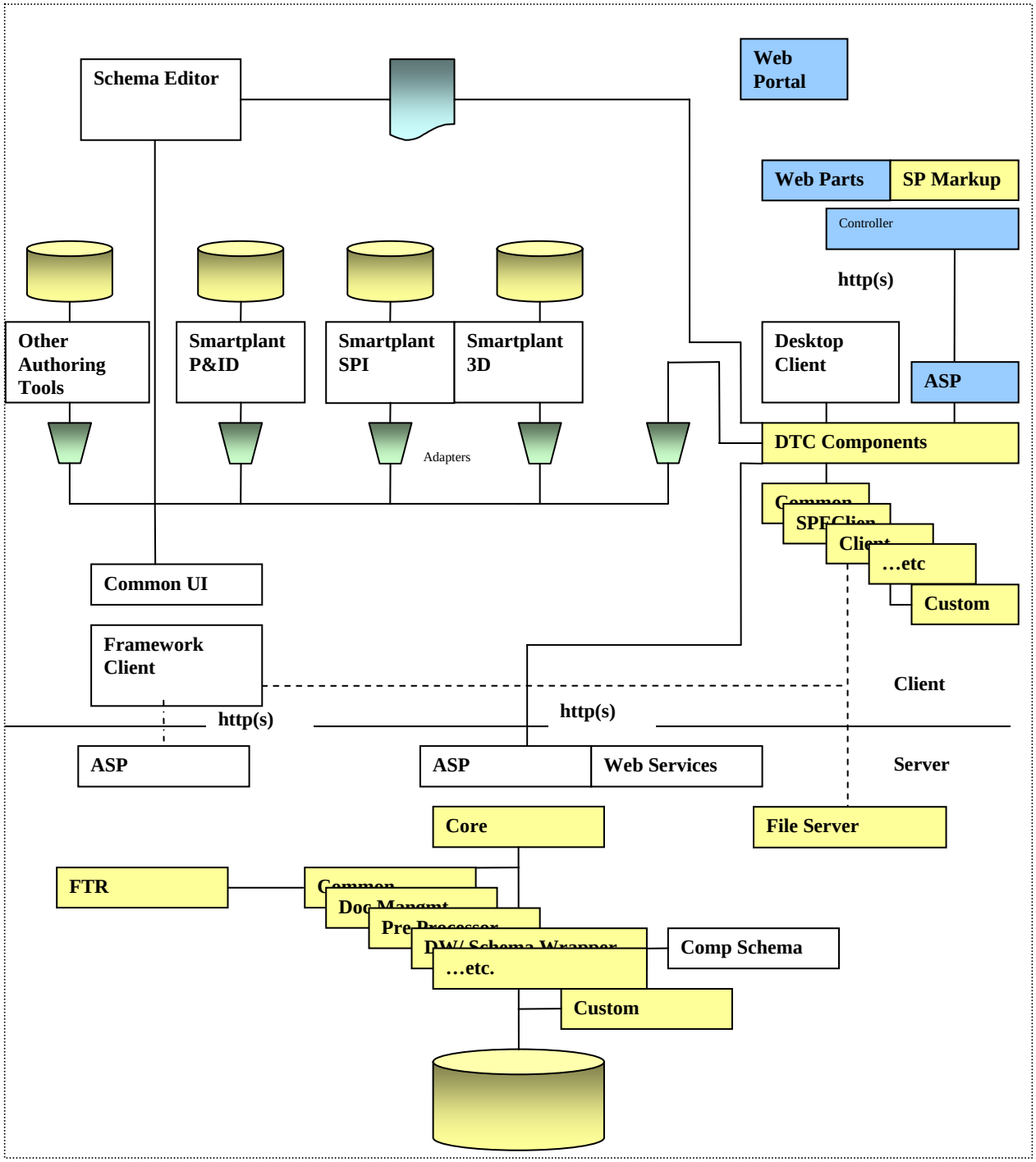
The *Compare* assembly needs the *Adapter* assembly in order to function.

On the SPF server side, a different set of assemblies get loaded. These assemblies are *DocMgmt*, *Progress*, *Load PP* (pre-processor for publish and retrieve), *TEF* (SP Integration), *SchemaScU* (SchemaCompProvider) and *SchemaComponent*. The SchemaCompProvider is a way of wrapping the Schema Component, which is a COM assembly. Schema Component gets wrapped in SchemaCompProvider, which is a .Net assembly in order to call it.



If a custom assembly needs to be loaded by the server, then it must be added to the *Settings.xml* file.

SPF 2008 Component Architecture



Unlike SPF 2007, which used two Oracle users (databases), SPF 2008 combines the admin configuration and the instantiated data objects into a single SPF database.

Introduction to SPF Architecture

The SPF core model is delivered with two databases:

- **SPF42 - The authoring database**
- **SPF42adw - The data warehouse database**

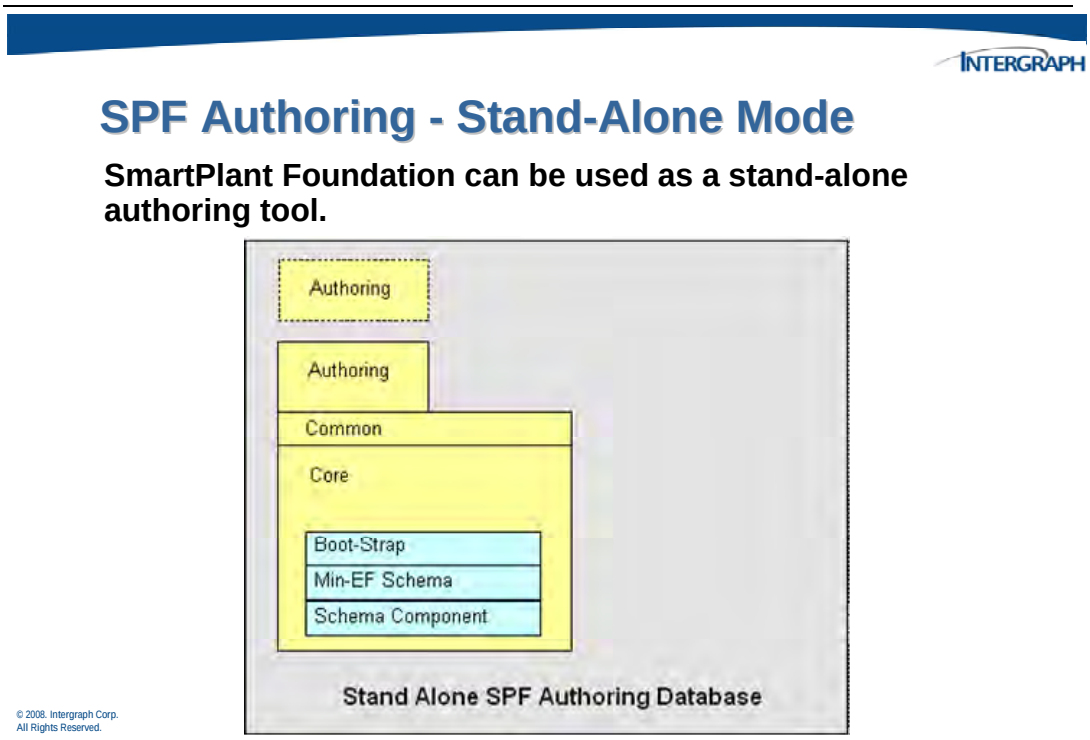
© 2008. Intergraph Corp.
All Rights Reserved.

Depending on which mode is specified during installation will determine which of these databases will be used.

1.1.1 SmartPlant Foundation Authoring - Stand-Alone Mode

In this mode, SmartPlant Foundation is used as a stand-alone authoring tool in which data and document meta data is created, modified and managed indigenously. This means that data is created and managed interactively using the *Desktop Client*, through the *Desktop Client Loader*, or from *Excel* using data sheets.

SmartPlant Foundation may publish data created in SmartPlant Foundation to other tools. However, it will not receive any data published from other authoring tools; that is, it would not operate in a data warehouse mode. The following block diagram depicts this model, including the various layers that are needed for SmartPlant Foundation to operate in this mode.



In this example, the block *Min EFSchema* refers to the minimum set of SmartPlant schema definitions on which the **Core** layer depends on. This block is mandatory for SmartPlant Foundation to work in stand-alone mode.

The **Boot-Strap** layer defines the minimum SmartPlant Foundation configuration to enable a user to launch the Desktop Client.

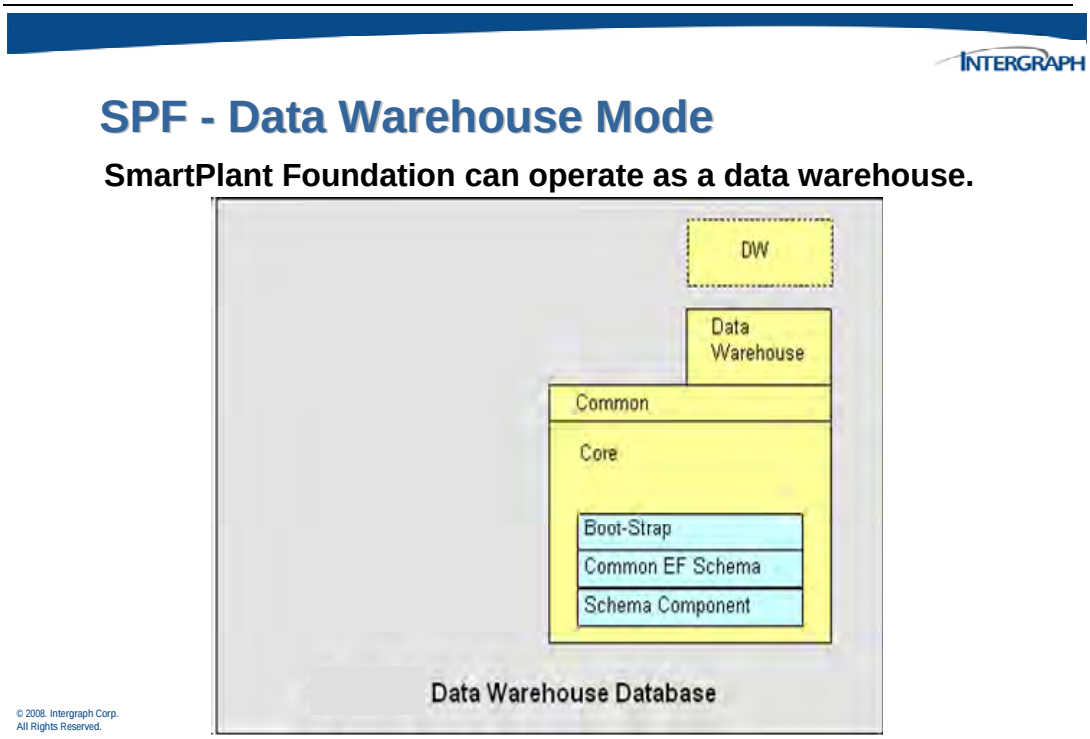
The **Core** adds the majority of the SmartPlant Foundation configuration, including all the administration classes and menus, template and design documents and support items like folders and organizations.

The **Common** layer defines the WBS structure, transmittals and vendor documents. These are not in the core because not all systems will use the same WBS or transmittal structure for example.

The **Authoring** layer defines all of the configuration to support using SmartPlant Foundation as a tool that can be integrated with the data warehouse. This includes functionality such as data sheets and data lists, etc.

1.1.2 SmartPlant Foundation - Data Warehouse Mode

In this mode, SmartPlant Foundation operates as a data warehouse. It operates as a repository to store and view published data coming from various authoring tools. In this scenario, the complete SmartPlant schema would be delivered in order to support the data warehouse layer.



In this diagram, the majority of the configuration is common to the authoring mode.

The **Data Warehouse** layer defines the configuration for supporting the storage, loading, correlation and navigation of data published by the design tools, including SmartPlant Foundation.

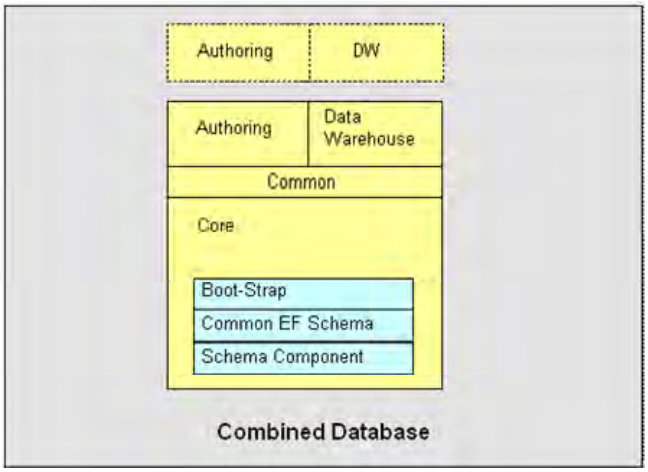
Configurations can use a separate SmartPlant Foundation server and database, if SmartPlant Foundation is to be also used in *Authoring* mode.

1.1.3 Combined Mode

In this mode, SmartPlant Foundation works with a single integrated database and caters to both SmartPlant Foundation **authoring and data warehousing** functions. The following block diagram depicts this mode.

SPF - Combined Mode

SmartPlant Foundation can work with a single integrated database and cater to both SPF authoring and data warehousing functions.



© 2008. Intergraph Corp.
All Rights Reserved.

1.2 SPF Containers

Part of the re-design of SPF 2008 is a feature where each object will have a *containerID* that will allow data definitions to be divided into sub-groups.

Containers

Containers segment components of the SPF configuration.

SPF is built from sets of files.

Each object and relationship *can* have a containerID, and is used to separate the data into sub-groups.

Usually, the following objects have containerIDs

- **ADMIN and SCHEMA (domains)**
- **Reference Data (for example, classification trees)**

Note: ContainerIDs can cross domains, thus Authoring appears in both the ADMIN and SCHEMA domains.

© 2008. Intergraph Corp.
All Rights Reserved.

A containerID should be set in order to specify the sub-grouping for these new definitions prior to creating new security objects.

Container ID Naming Convention

ContainerIDs are made up of the following:

Layer.Container

where

Layer = directory

Container = prefix_filename

(prefix = Admin, Schema or Domain)

© 2008. Intergraph Corp.
All Rights Reserved.

Examples of containerIDs are provided below

- Bootstrap
- Core
  - Core.Forms
  - Core.Reports
  - Core.Workflow
- Common
  - Common.Documents
  - Common.Transmitals
- Authoring

INTERGRAPH

Containers and Model File Layers

The system ContainerIDs are often common across schema, admin, and data components. To keep them unique, they are appended with SCHEMA, ADMIN, or DATA.

SchemaComponent

- **Contains all base EF schema required by all SmartPlant Foundation systems**

Bootstrap

- **Contains all the necessary schema and administration items necessary to run the Desktop Client.**

Core

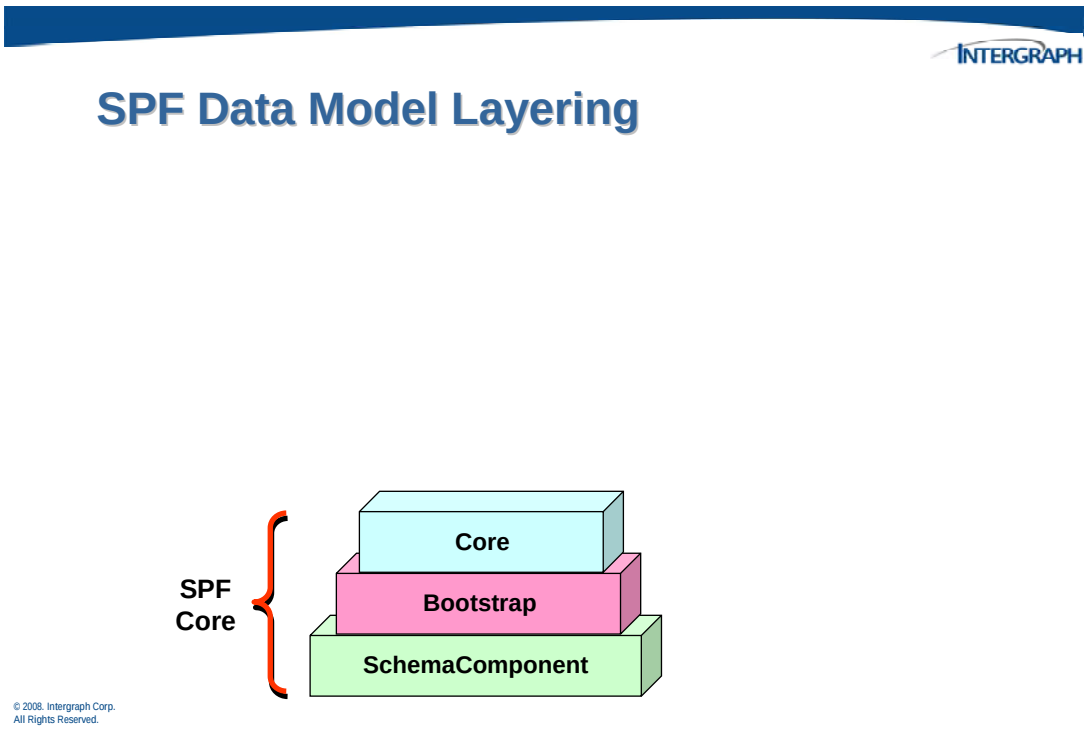
- **Built on the *Bootstrap* and *SchemaComponent* layers and provides the minimum configuration to run SmartPlant Foundation.**

© 2008. Intergraph Corp.
All Rights Reserved.

1.3 Data Model Layering

The delivered model files are normally located in \Program Files\SmartPlant\Foundation\2008\Models.

Many of these directories contain configuration files and sub directories for schema diagrams. The root directory contains model definition files (.xmlsrc) that control which sets of model files are to be built into a database.

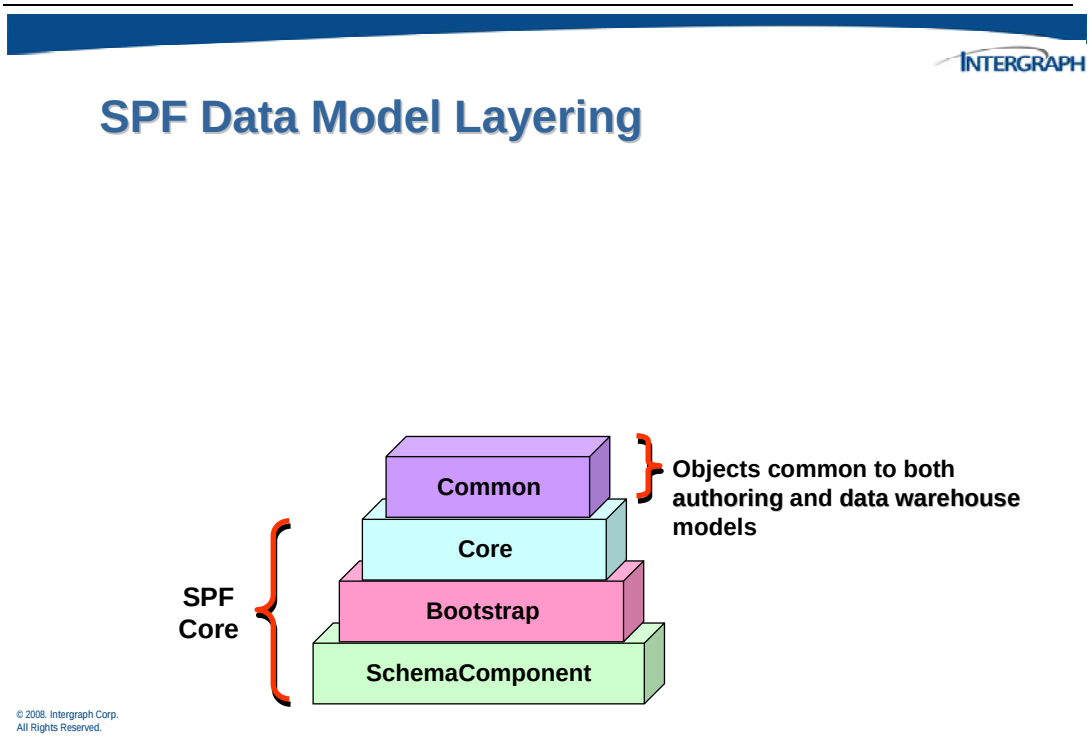


The SPF Core or startup layer consists of the following layers:

- **SchemaComponent Layer** - The container ID for this is *SchemaComponent.SchemaComponent*. It contains all base EF schema required by all SmartPlant Foundation systems.
- **Bootstrap Layer** - The container ID for this is *Bootstrap.Bootstrap_SCHEMA* and *Bootstrap.Bootstrap_ADMIN*. It contains all the necessary schema and administration items necessary to run the Desktop Client. The administration items provide access to minimal commands, such as run the loader or exit. The classes that these manage are configured in the Core layer.
- **Core Layer** - This layer is built on the *Bootstrap* and *SchemaComponent* layers and provides the minimum configuration to run SmartPlant Foundation. It does not deliver the plant and project classes because these are not truly generic and a non-integrated system can have any configuration hierarchy.

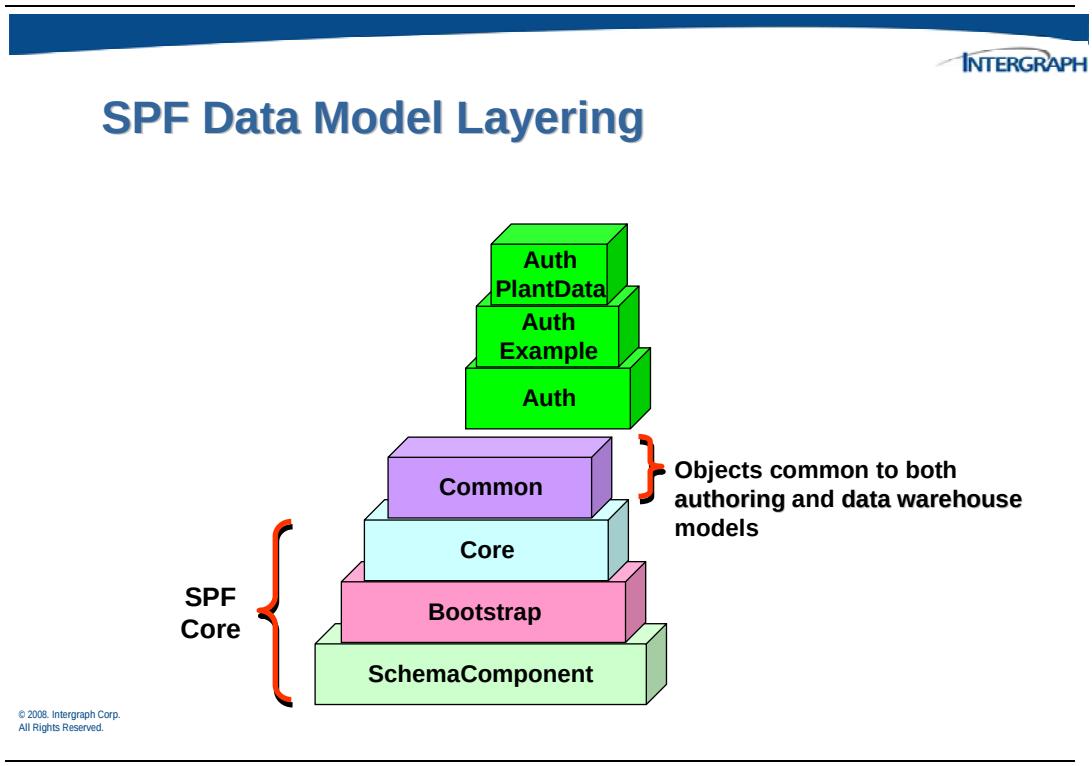
There are many containers in this layer: some to define the schema, some to define internal code components, and some to administer the system and security configuration.

The order of the containers reflects their dependency.



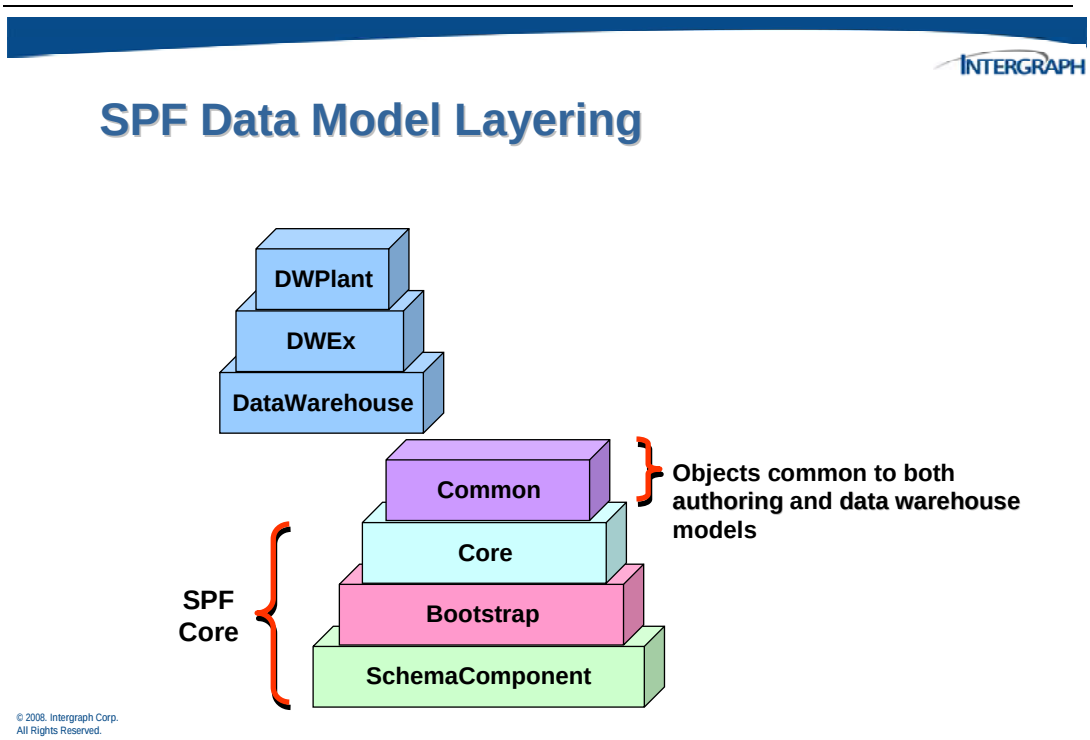
The following additional layers are used to build the SPF databases.

- **Common** - This layer contains the definition of objects that are common to both the authoring and data warehouse models. They are separated from the core because the exact structures may change from one installation to another. These can be selectively changed and a new database build using SPF administrator. The following configuration is found in this layer:
  - WBS: Plant and Project
  - Internal and external transmittals
  - Vender documents



Additional layers (con't)

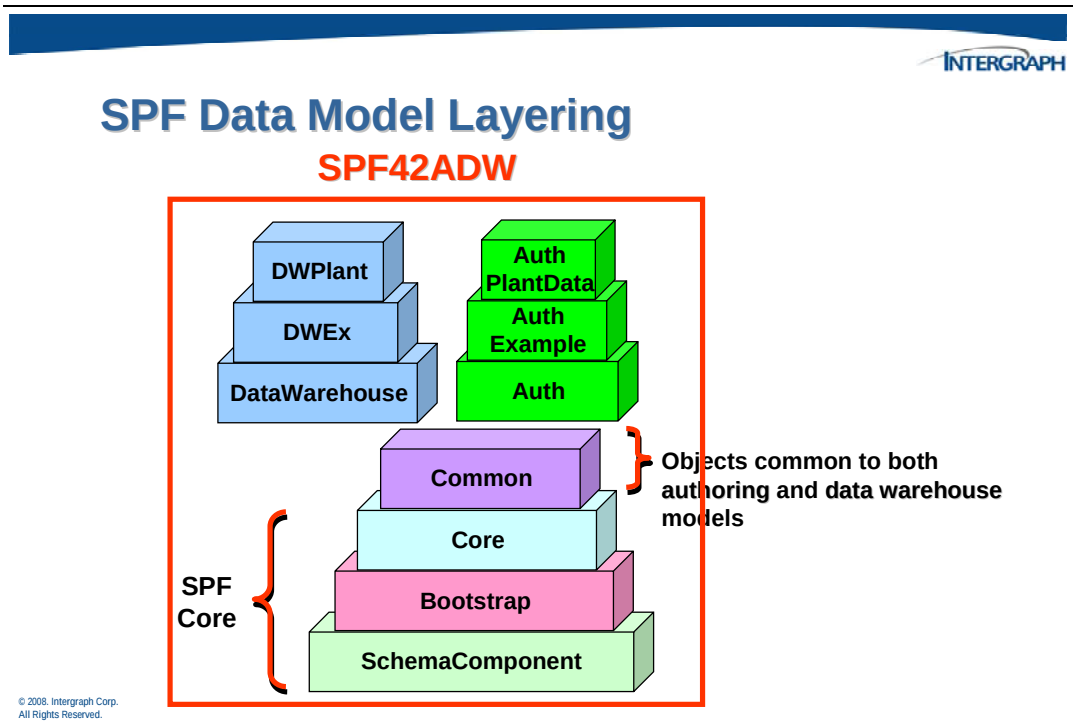
- **Auth** - This layer contains the definition of scheme and configuration for mechanical and process engineering using data sheets and data lists. It also sets up the progress hierarchy structure.
- **AuthExample** - This layer contains the definition of the example data sheets.
- **AuthPlantData** - This left contains the definition of example roles, users and example plant data.



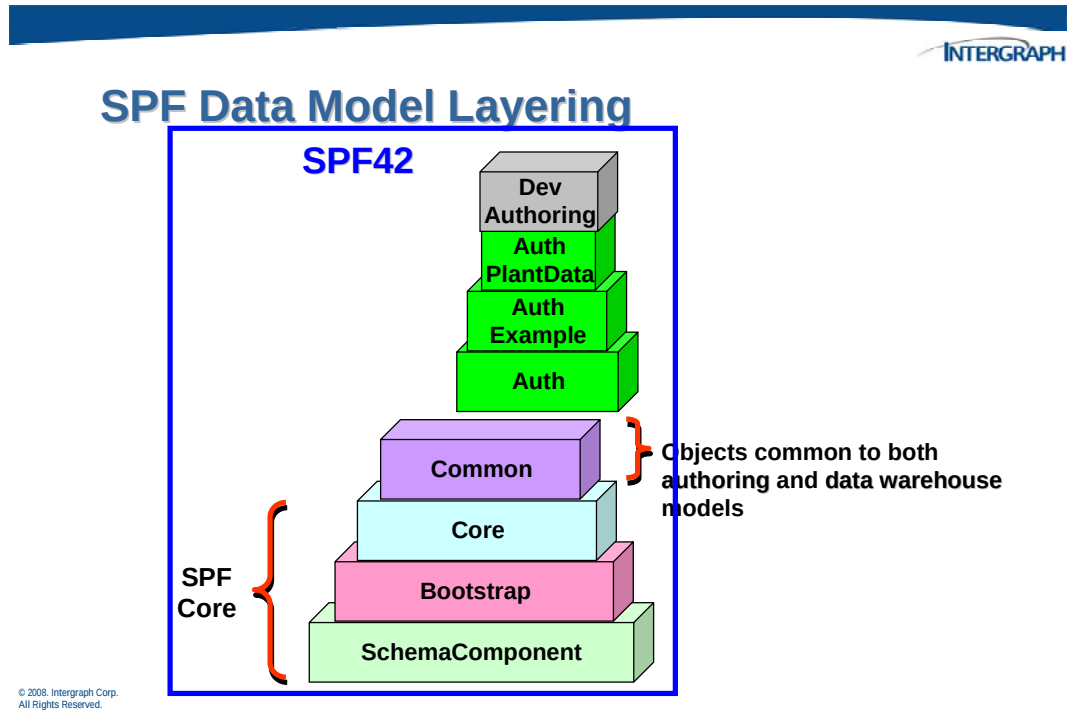
Additional layers (con't)

- **DataWarehouse (DW)** - This layer contains the schema and administration configuration required to manage the Data Warehouse including workflows, scheduler tasks, title block generation and data navigation.
- **DWEx (DWExample)** - This layer contains some sample EdgeDefs for relationship navigation.
- **DWPlant (DWPlantData)** - This layer contains the functional unit and area plant breakdown structure. It also contains the definition of example roles, users and example plant data.

The DataWarehouse model contains all of the data definitions to support the integration between SPF and the authoring tools.



The **SPF42ADW** delivered database contains both the *Authoring* and *Data Warehouse* layers as well as the *core* and *common* layers.



- **DevAuthoring** - This layer contains additional class definition and administration options to assist in the testing and certification of the product. The source files

provide good examples to a system developer. This layer is **not loaded** in the out of the box database.

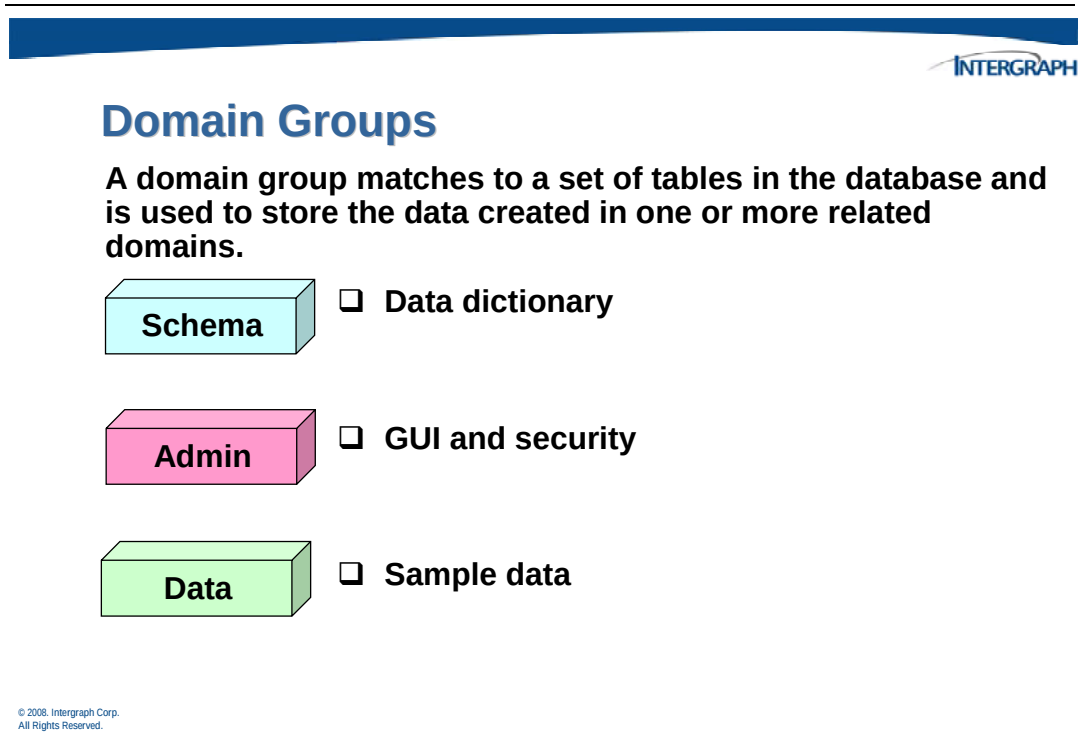
The DevAuthoring model **does not** contain any of the definitions to support integration.

1.4 Domains and Domain Groups

If there is a need to segregate data then **Domains** are a good solution. A component schema is linked to a domain so that all ClassDefs for that schema component are created in that Domain. Note that it is not possible to store objects of the same ClassDef in different domains.

A separate Domain is used for each tool publishing data into the data warehouse. Another use of a Domain may be to segregate human resources or management documentation.

An additional explanation of domains is included later in this chapter.



If a new domain group is required, there are database groups available for generating these database tables.

1.4.1 SPF 2008 Database Tables

The SPF database consists of 5 tables for each domain group. For example, in Combo mode, the SPF42AWD database would have a minimum of 20 tables, 5 for each domain group.

SPF Database tables

The database has been split 3 ways:

- **SCHEMA** — **Schema & Admin**
- **DATA** — **Authoring**
- **PUBLISH** — **All published data**

This design allows for:

- **Easier database tuning**
- **Data segregation**
- **Configurable security**

© 2008. Intergraph Corp.
All Rights Reserved.

The **Schema** domain group tables will contain the schema definitions and administration configuration objects.

SPF Database tables

There are 5 tables per domain group. The *Schema domain group* contains the tables for the Schema domain and the Admin domain.

- **SchemaObj** — ➢ ***Schema* and *Admin* object data (UID, Config, DateCreated, etc.)**
- **SchemaObjIF** — ➢ **Interface data for schema and admin objects**
- **SchemaObjPR** — ➢ **Property data for for schema and admin objects**
- **SchemaObjRel** — ➢ **Relationship data for schema and admin objects**
- **SchemaObjPRExt** — ➢ **Property blob data for schema and admin objects (when > 4000 chars)**

© 2008. Intergraph Corp.
All Rights Reserved.

The **Data** domain group tables will contain authoring objects.

SPF Database tables

These 5 tables are repeated for the *Data domain group*.

- **DataObj** — ➢ **Authoring object data (UID, Config, DateCreated, etc.)**
- **DataObjIF** — ➢ **Interface data for authoring objects**
- **DataObjPR** — ➢ **Property data for authoring objects**
- **DataObjRel** — ➢ **Relationship data for authoring objects**
- **DataObjPRExt** — ➢ **Property blob data for authoring objects (when > 4000 chars)**

© 2008. Intergraph Corp.
All Rights Reserved.

The example below shows the contents of the database tables. This is repeated for each domain group.

SPF Database Tables

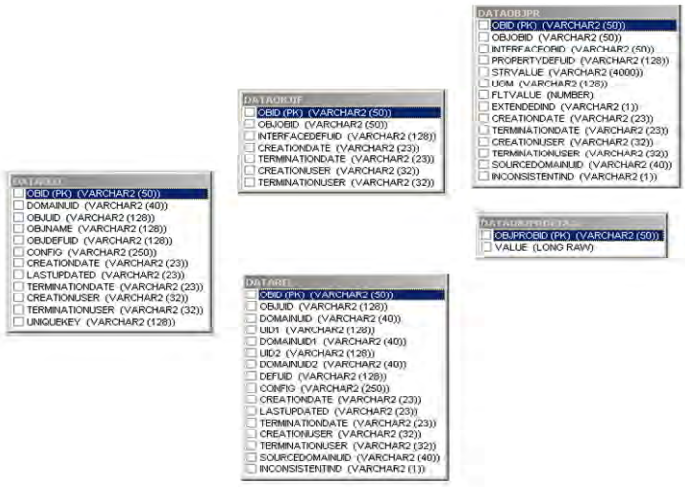
DataObj

DataObjIF

DataObjPR

DataObjPRExt

DataRel



Schemaxxx

Pubxxx

Oracle & SQLServer

© 2008. Intergraph Corp.
All Rights Reserved.

The **Published** domain group tables will contain published (data warehouse) objects.

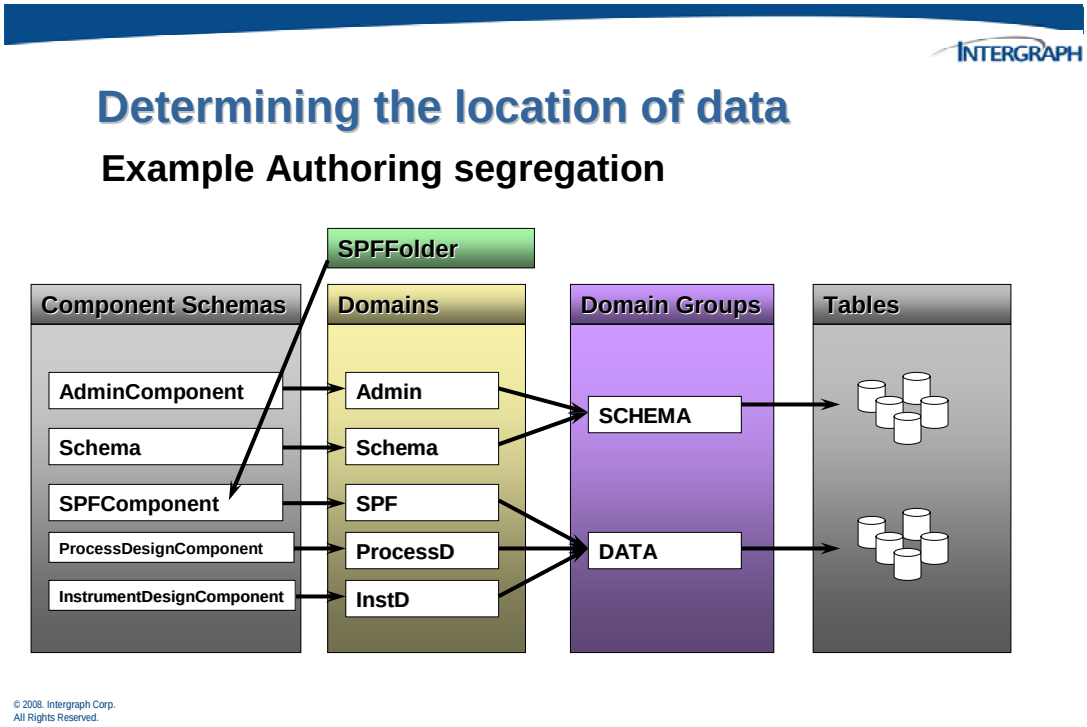
SPF Database tables

These 5 tables are also repeated for the *Published domain group*.

- **PubObj** — ➢ **Published object data (UID, Config, DateCreated, etc.)**
- **PubObjIF** — ➢ **Interface data for published objects**
- **PubObjPR** — ➢ **Property data for published objects**
- **PubObjRel** — ➢ **Relationship data for published objects**
- **PubObjPRExt** — ➢ **Property blob data for published objects (when > 4000 chars)**

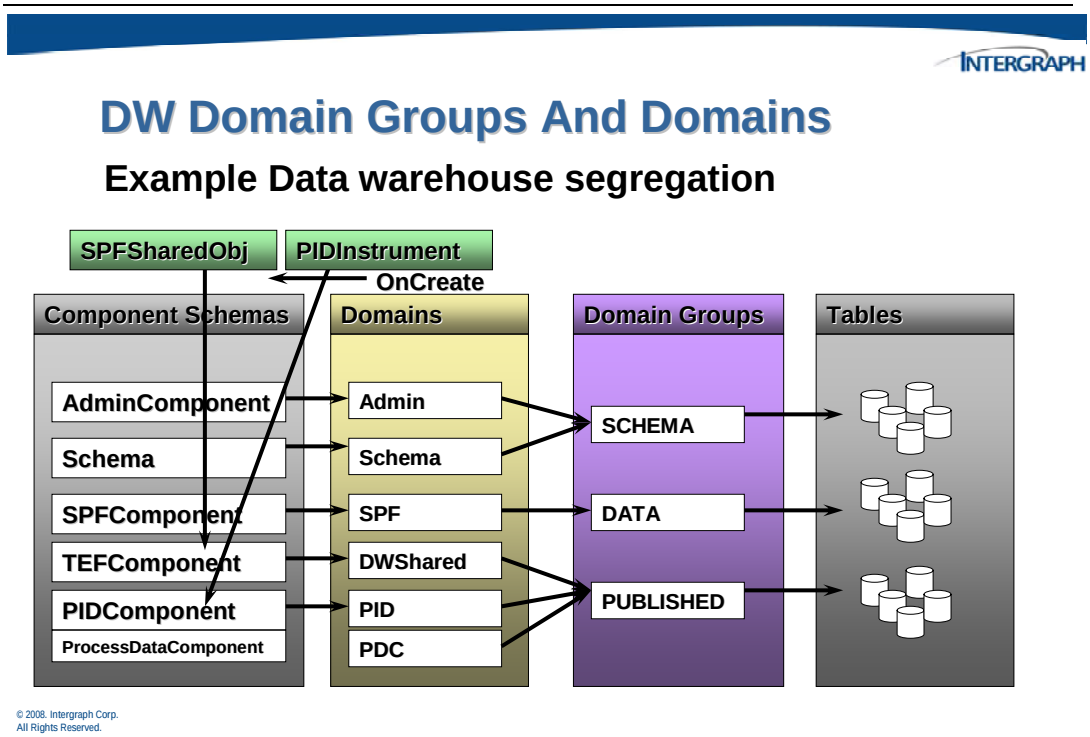
© 2008. Intergraph Corp.
All Rights Reserved.

The relationship between component schemas, domains and domain groups will determine which tables will contain the instantiation of object data.



In the above example, if a folder object instance is created, the relationship between the **SPFFolder** class def and the **SPFComponent** component schema will restrict the data instance to the *SPF* **domain**. The relationship between the SPF domain and the **Data** domain group would specify which database tables would contain the folder instance.

In the below example, if a **PIDInstrument** object instance is published, there is a performs relationship to a code override (MethodDef) called *OnCreate*. This is a software method the gets "fired" when an instrument is published to create a shared instrument object.

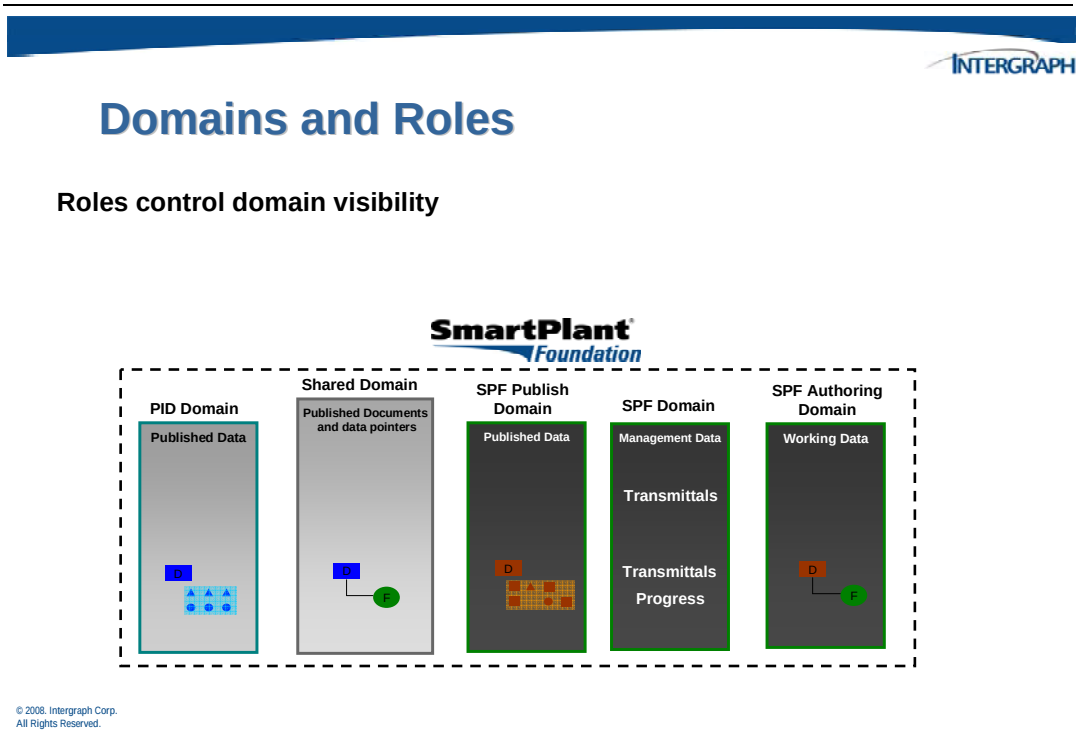


Again, the relationship between the PIDInstrument class and the **PIDComponent** component schema will restrict the PID instrument data to the *PID* **domain**. The relationship between the **SPFSharedObj** class and the **TEFComponent** component schema will restrict the shared instrument data to the *DWShared* **domain**. The relationship between the DWShared domain, the PID domain, and the **Published** domain group would again specify which database tables would contain the instrument instance.

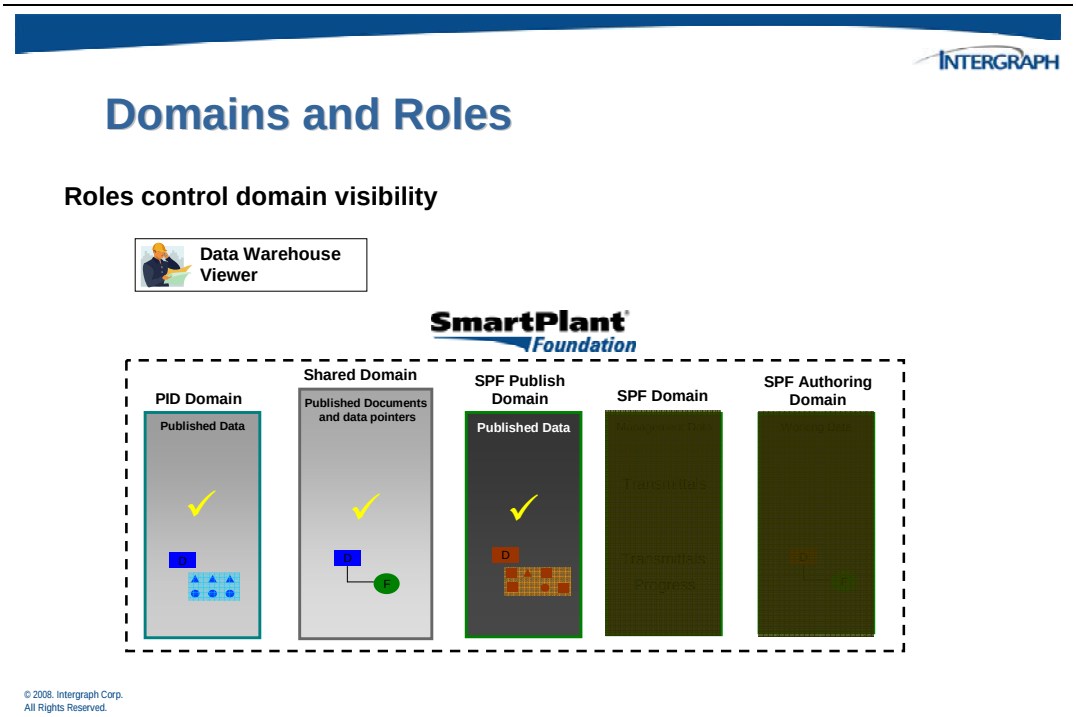
1.5 Overview of Domains and Roles

Domains provide data segregation and separate the data published by the tools. They also separate the working and published data in SPF.

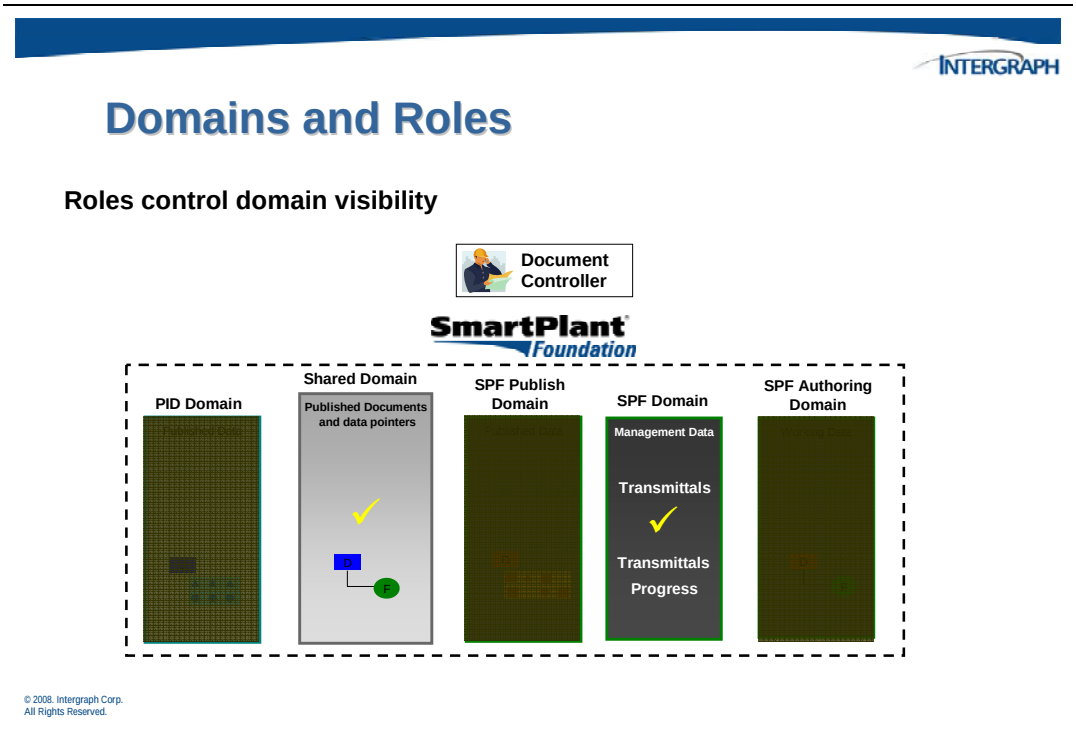
Roles control domain visibility and the level of access within domains.



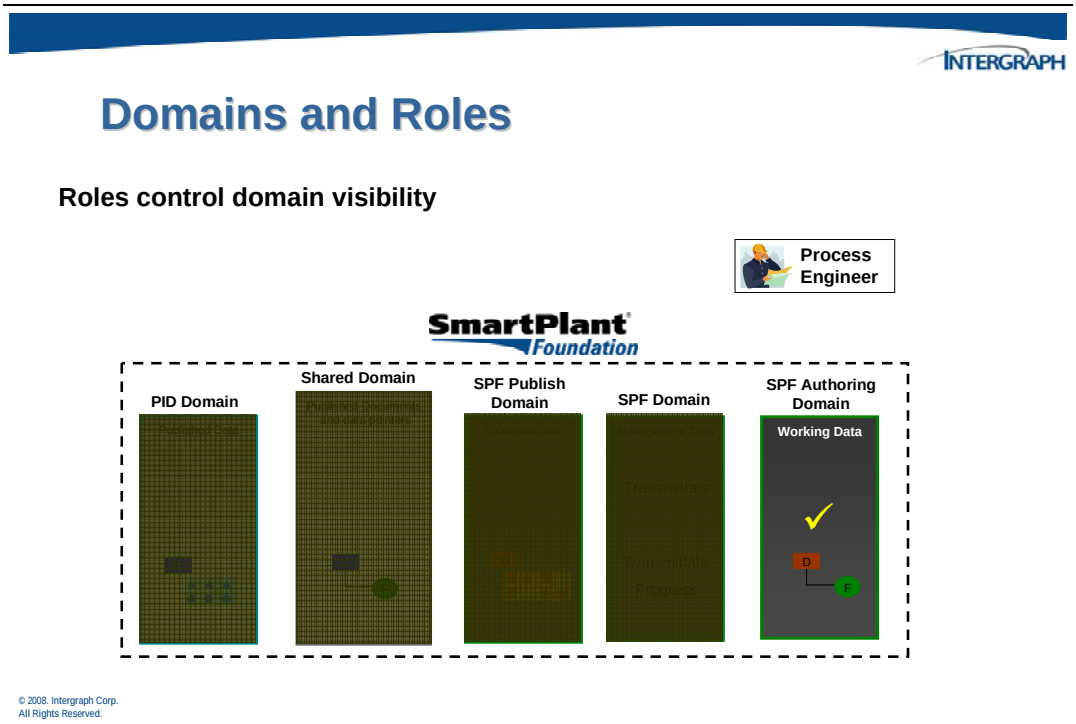
Role assignments control access within Plants. A user assigned to the role of **Data Warehouse Viewer** would be limited to the *PID domain*, the *Shared domain* and the *SPF Publish domain*.



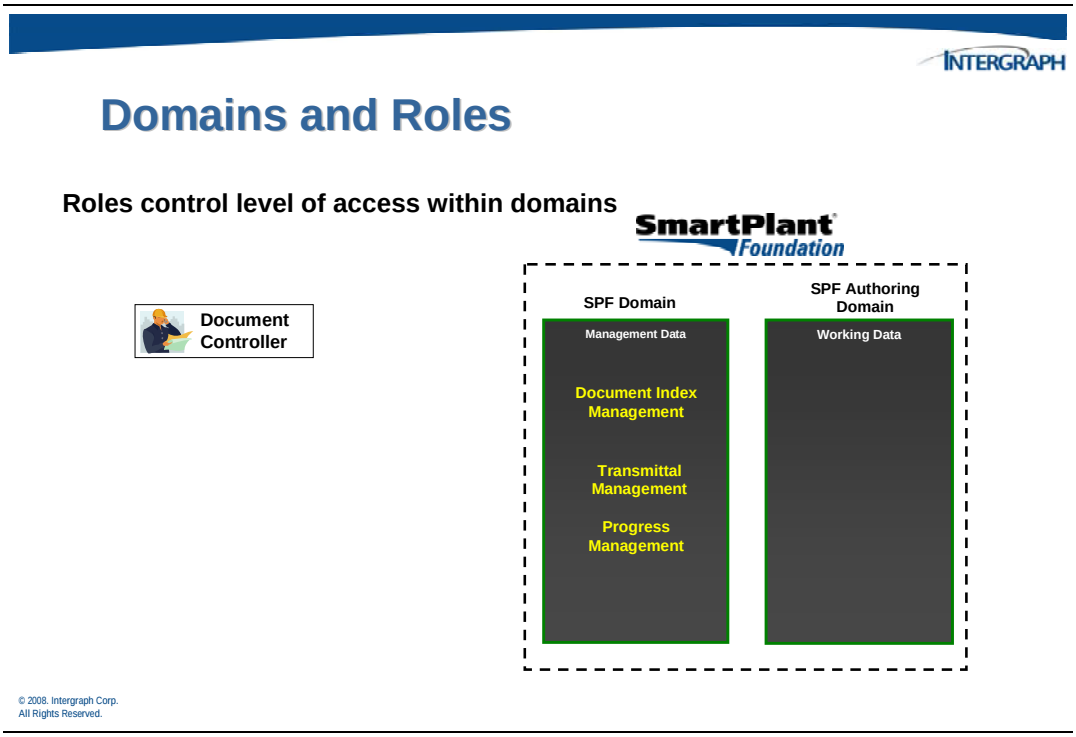
On the other hand, a user assigned to the role of **Document Controller** would have access limited to the *SPF domain*.



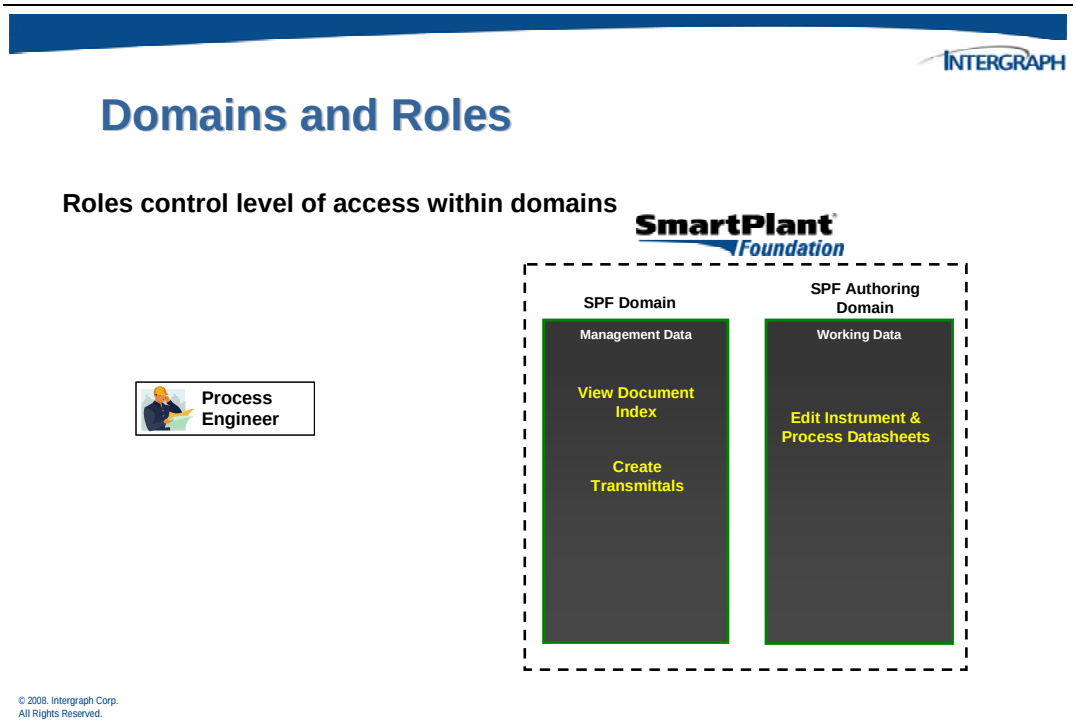
A user assigned to the role of **Process Engineer** would have access limited to the *SPF Authoring domain*.



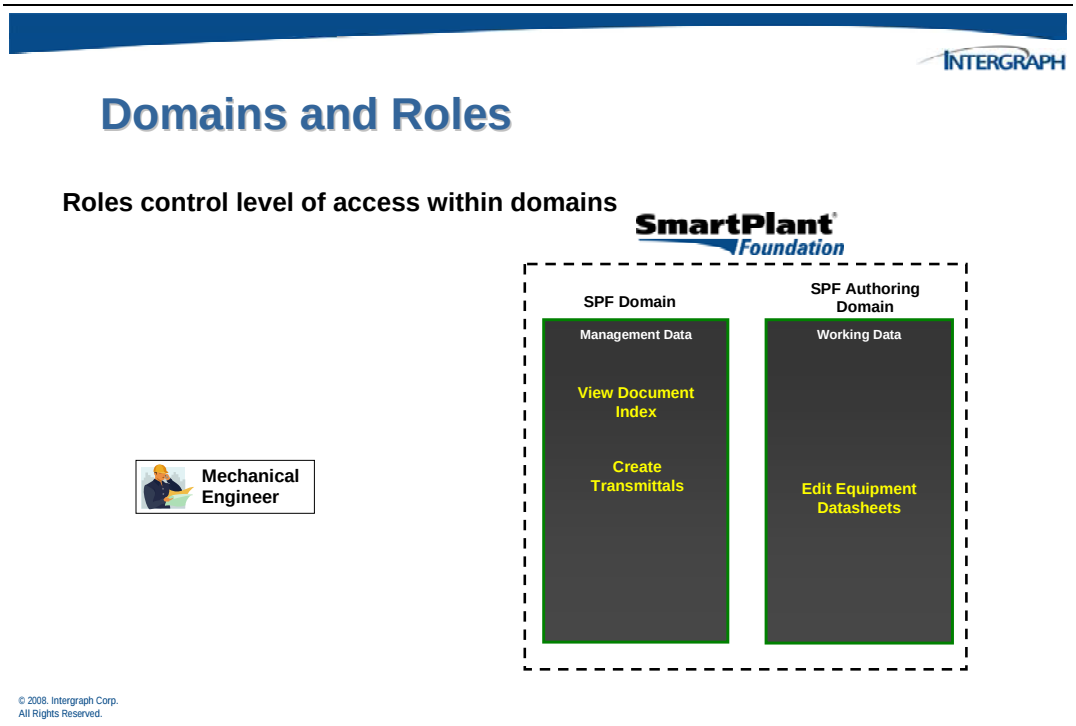
User functionality can be controlled within a domain. A *Document Controller* user would be restricted to functions in the SPF domain.



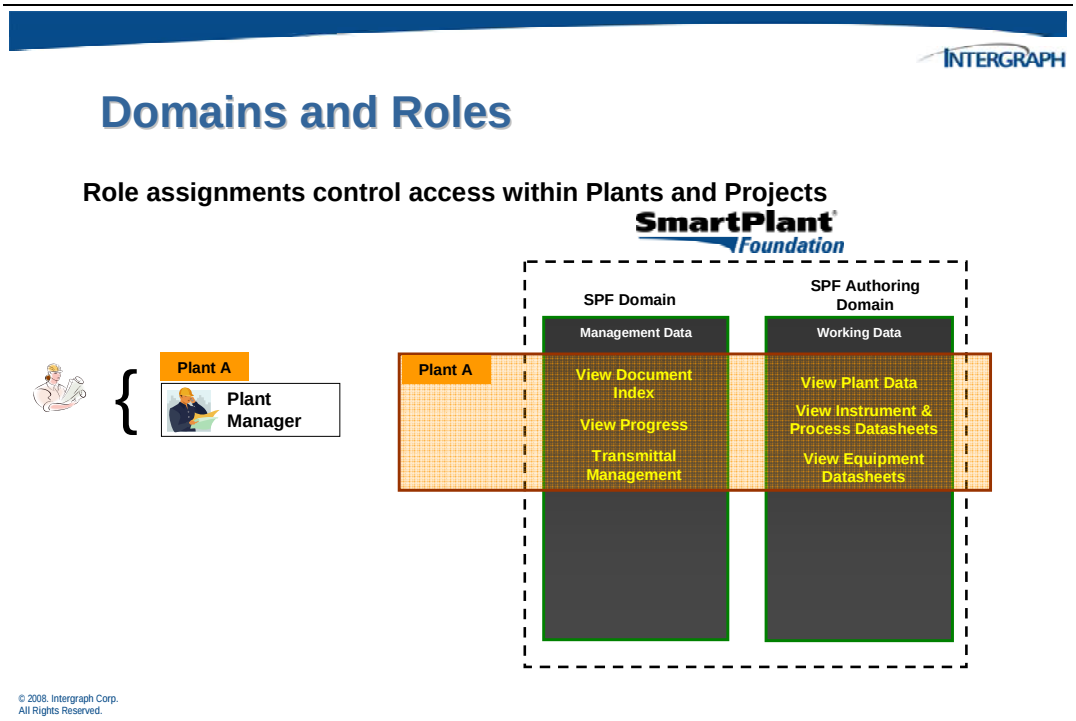
A *Process Engineer* user would be allowed access to the SPF domain AND the SPF Authoring domain.



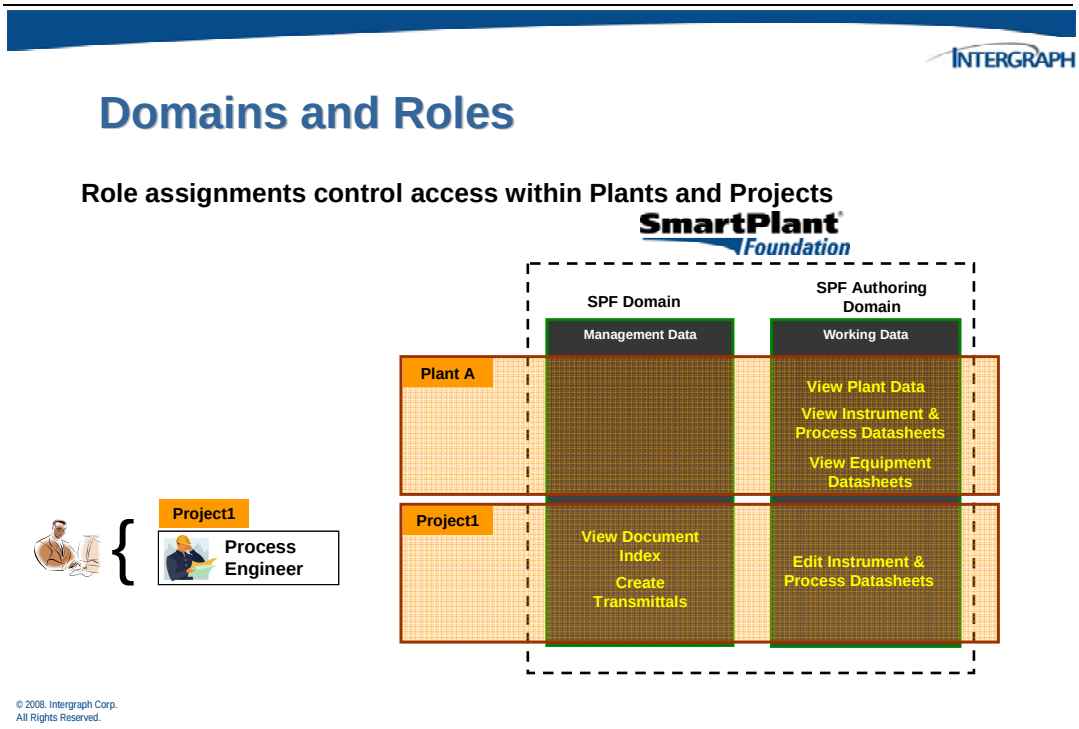
A *Mechanical Engineer* user would be allowed access to **different** functions in both the SPF domain and the SPF Authoring domain.



In addition to controlling domain access, role assignments give users access to plant data in all domains visible to that role



Other role assignments allow for user access to both plant and project data across visible domains.

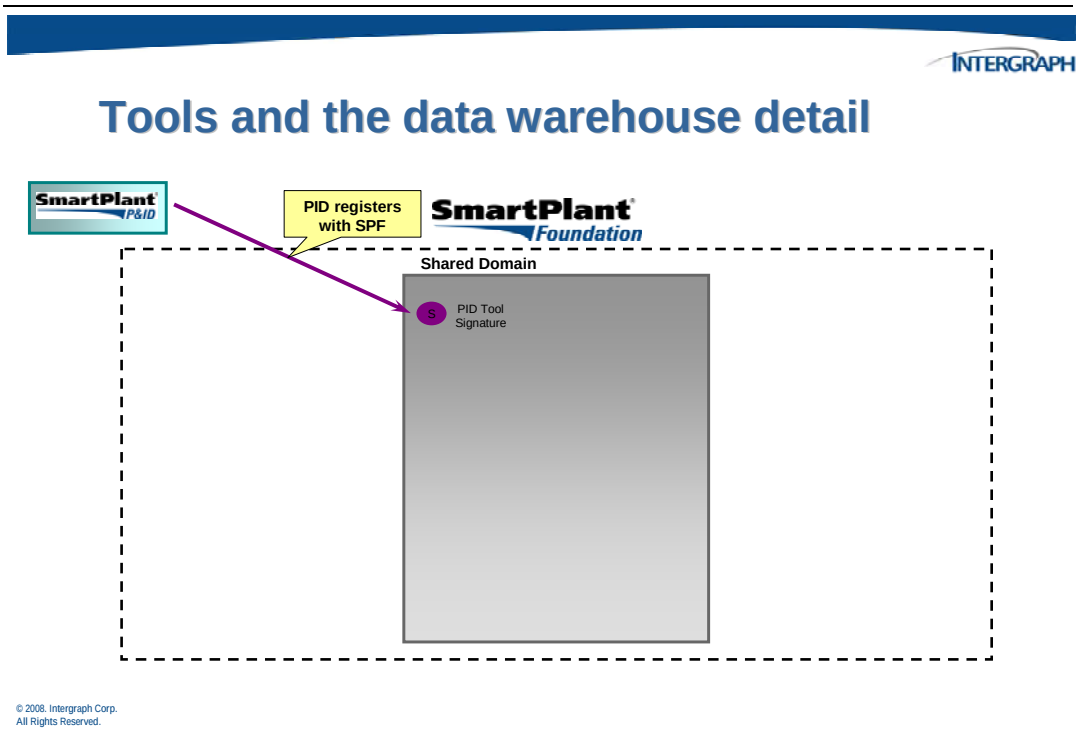


1.6 Data Warehouse Overview

SmartPlant Foundation (SPF) can be set up to behave like any other authoring tool; that is SPF can be set up to *edit*, *publish* or *retrieve* data to another SPF system (or in some cases the same SPF system) acting as a data warehouse.

With the use of domains, it is possible to set up one instance of SmartPlant Foundation to behave as a tool and as a data warehouse. Domains allow for actual separate data storage, not just a database of pointers. Each domain group is a set of Oracle tables.Tool Publishing Using Domains

Before publish or retrieve can take place, the authoring tool will need to register itself with the data warehouse.

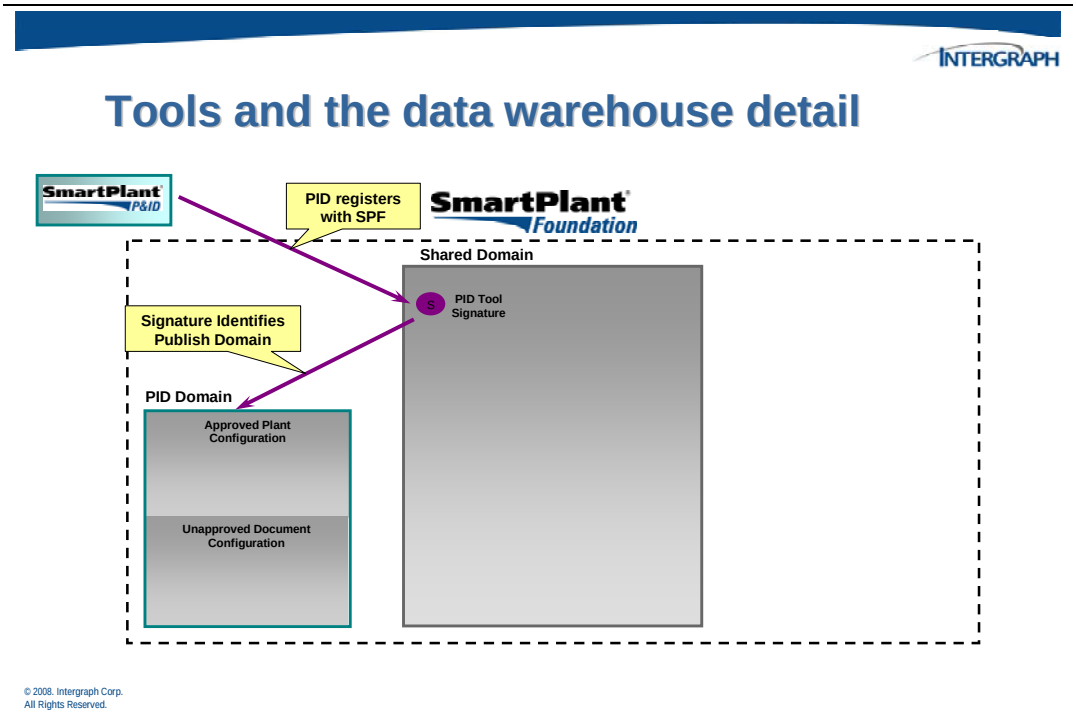


The SmartPlant Registration maps the authoring tool's plant and all its projects to a SmartPlant Foundation URL, which points to one SmartPlant Foundation plant, and returns a unique signature for the tool and plant combination being registered.

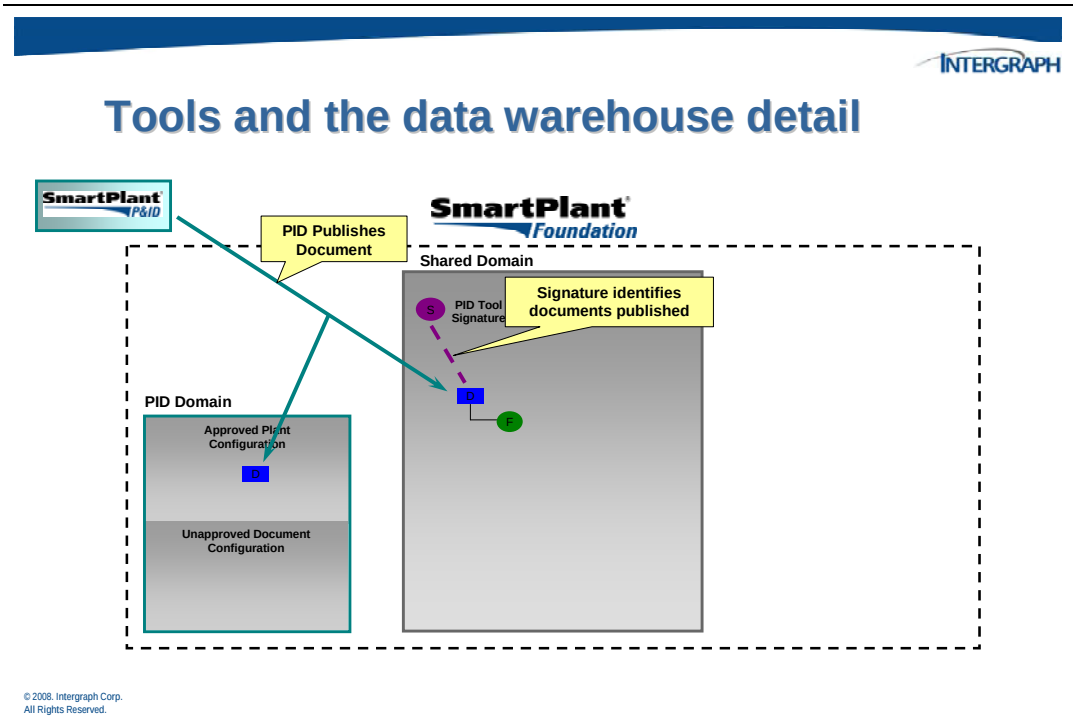
Each plant in the authoring tool must be registered at least once before documents can be published or retrieved.

The registration process also compares the authoring tool’s schema release number against the list of supported release numbers on the SmartPlant Foundation server. If the tool schema is compatible, the tool is granted registration.

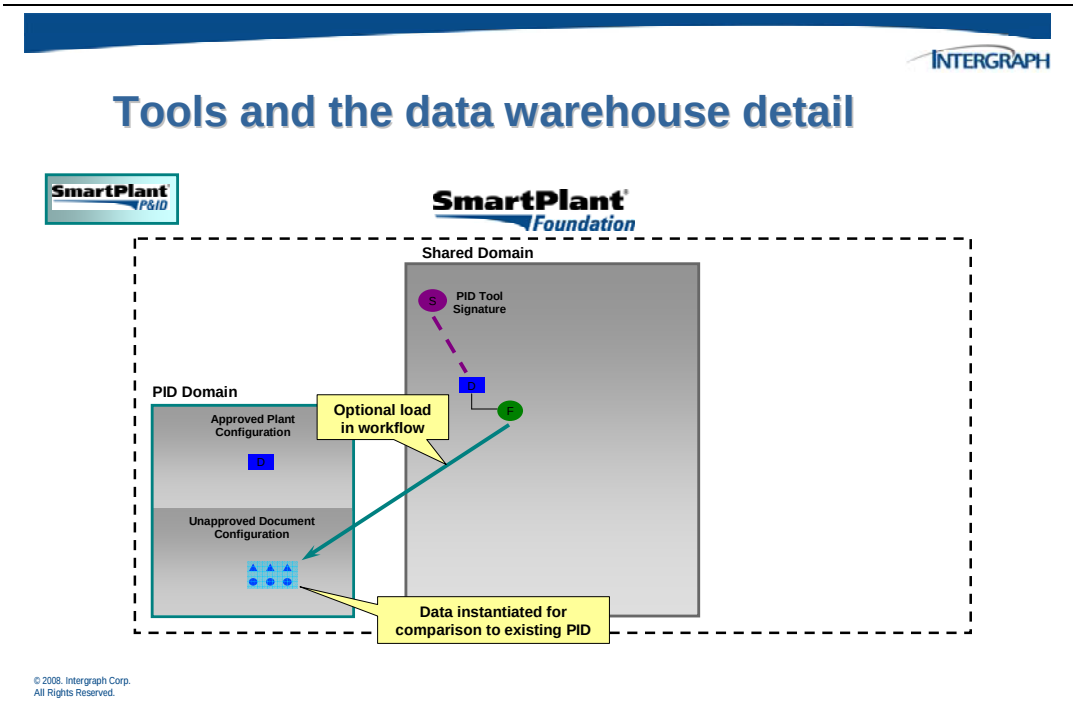
When the registration operation is performed, a signature is generated in the data warehouse and returned to the authoring tool. Within authoring this signature is saved as a registration record to be used for subsequent publish and retrieve operations.



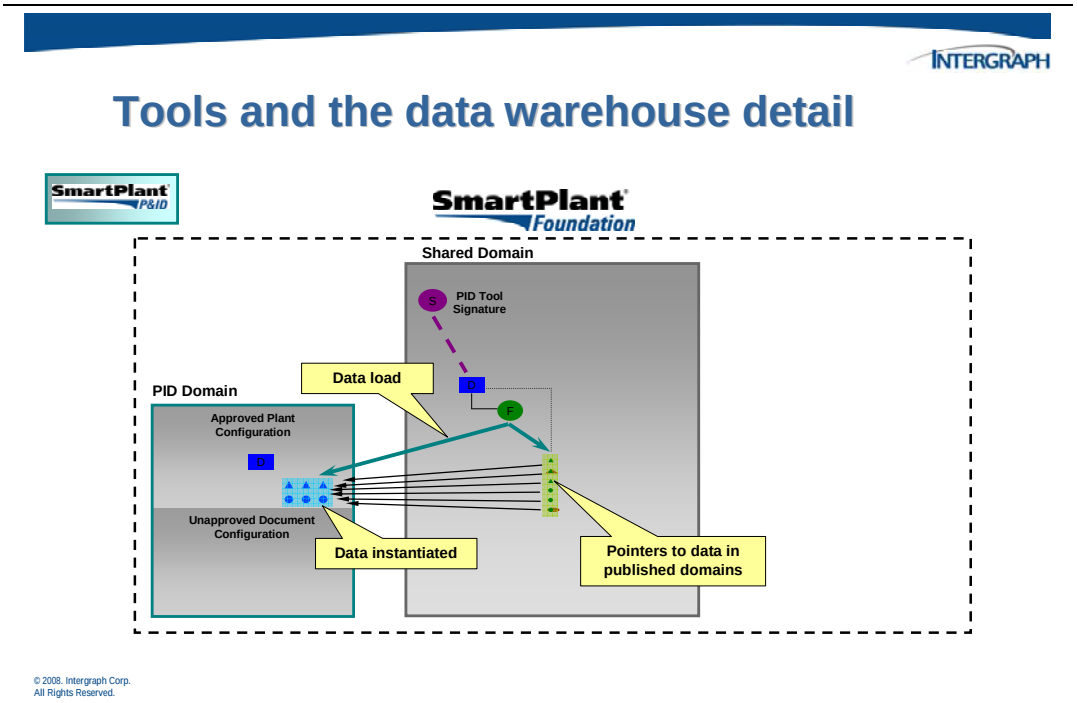
When a tool publishes data, then a document is created in the Shared domain in the specified configuration. Any files that are associated are also attached in this domain.



At this stage the document is "unapproved". As part of the workflow, there is an optional step to load the data into a sub configuration in the tool domain. In here the data can be viewed and navigated and then approved (or rejected).



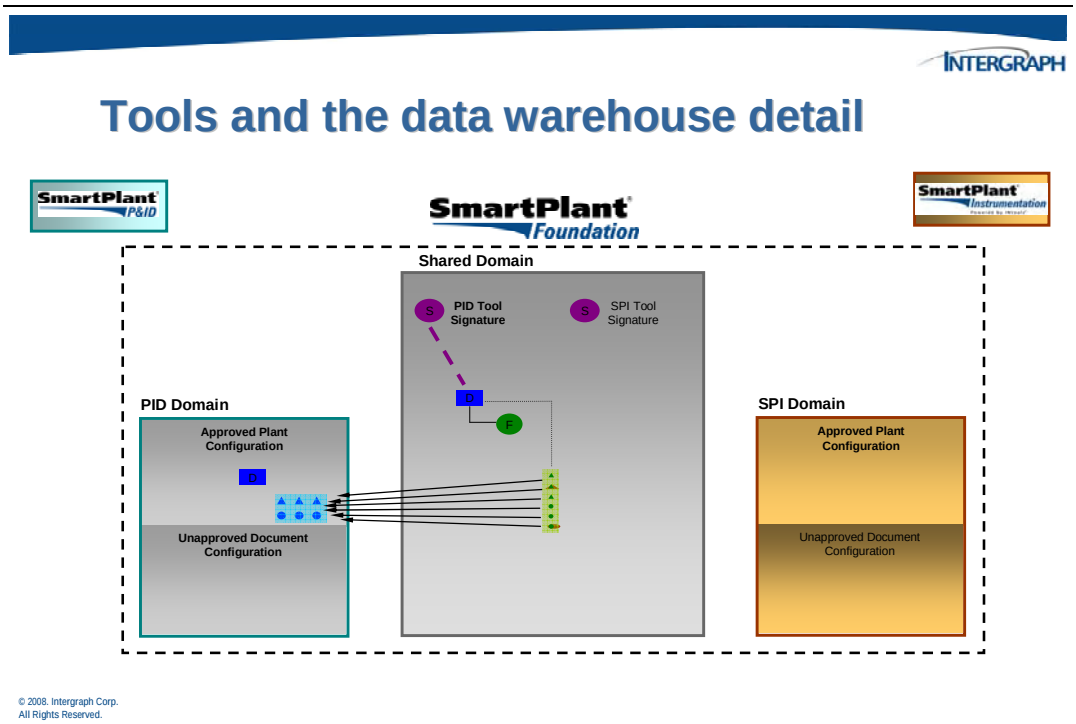
After approval, the data contained in the associated files is loaded into the necessary tool domain.



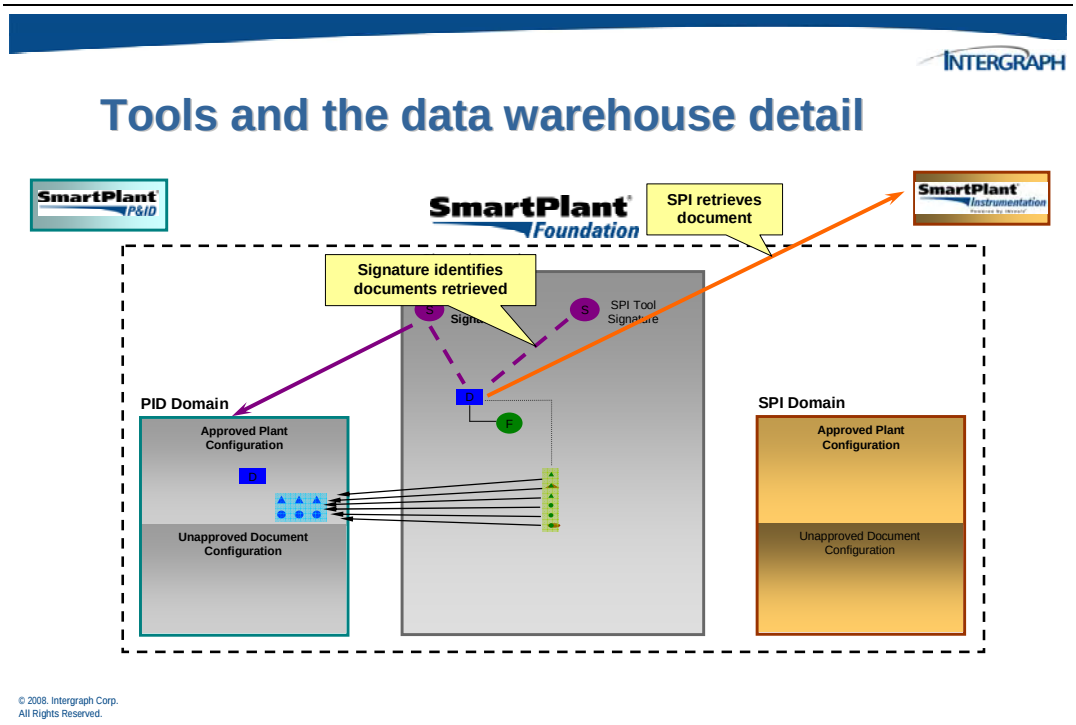
Whenever data is consolidated into a tool domain, such as PID, a proxy object (or shared object) is created in the central Shared domain. This object will only instantiate one or two key interfaces and rely on the relationship (SPFComprisedOf) to point to the real data.

In the case of the document object, this is created as an object in the tool (as well as being represented by a 3 tier document) and rather than being associated to a proxy object this is related to the document master. The document master will also be related to the tool signature, which in turn is related to the plant and tool that the tool signature represents.

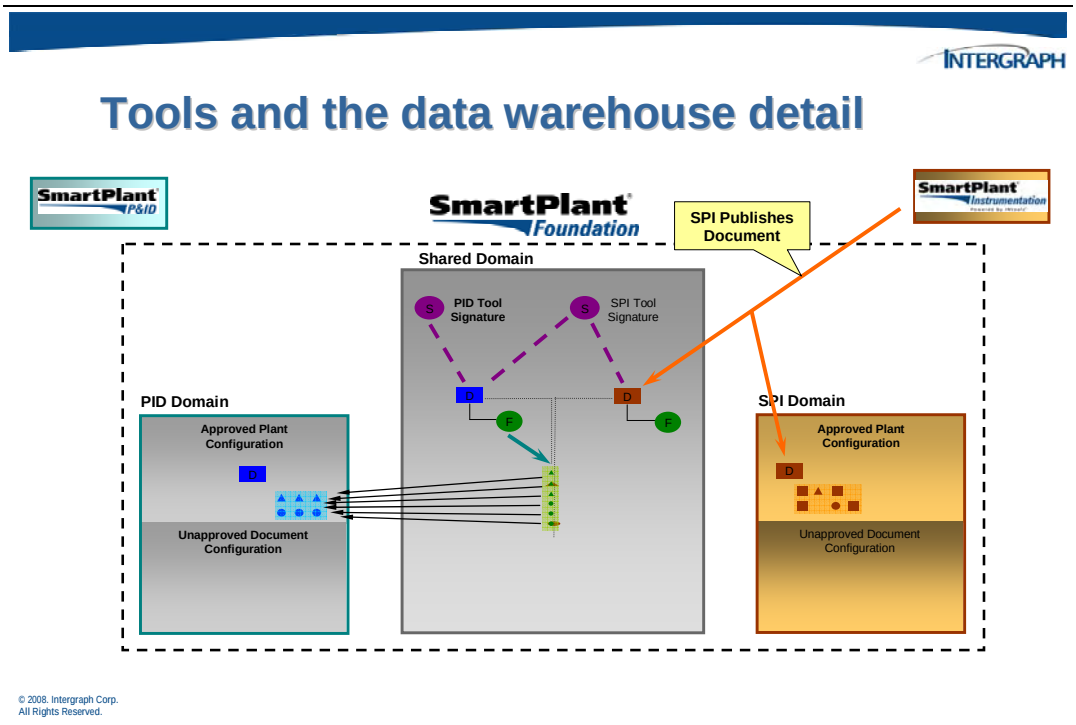
For each tool that publishes data to the data warehouse, a separate domain is needed. Each authoring tool, SPI example, will need to register itself with the data warehouse.



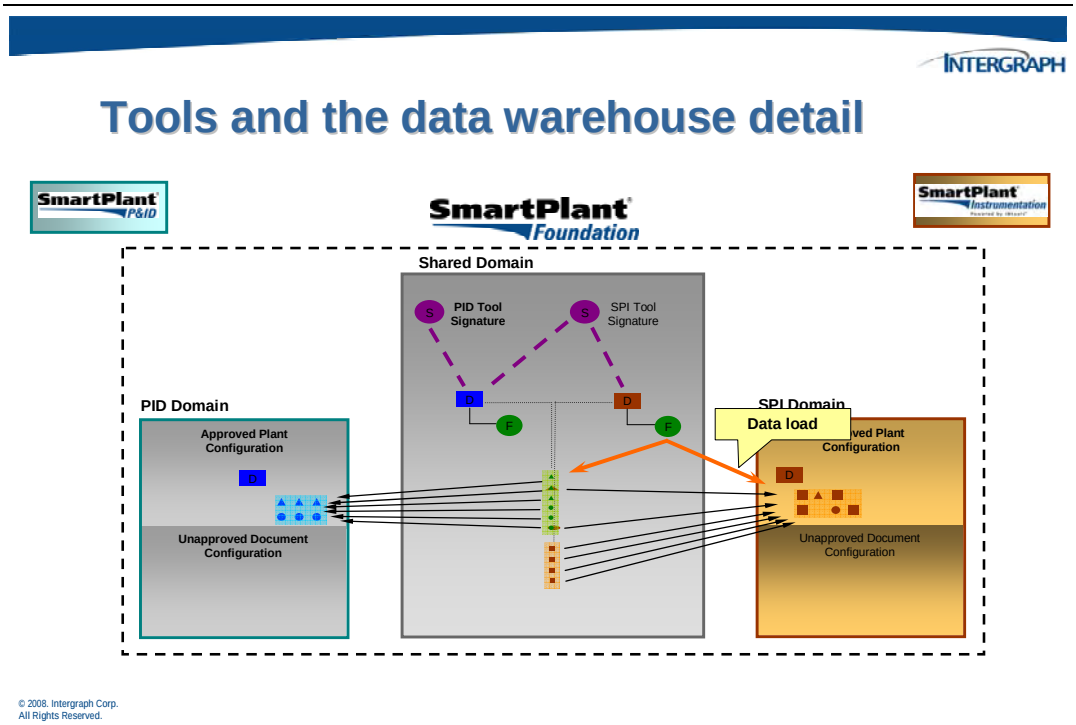
Once approved, a document can then be retrieved by another tool. For existing tool adapters, data is retrieved into a *To Do List* in the tool.



The process of publish can then be repeated once any design changes are complete.

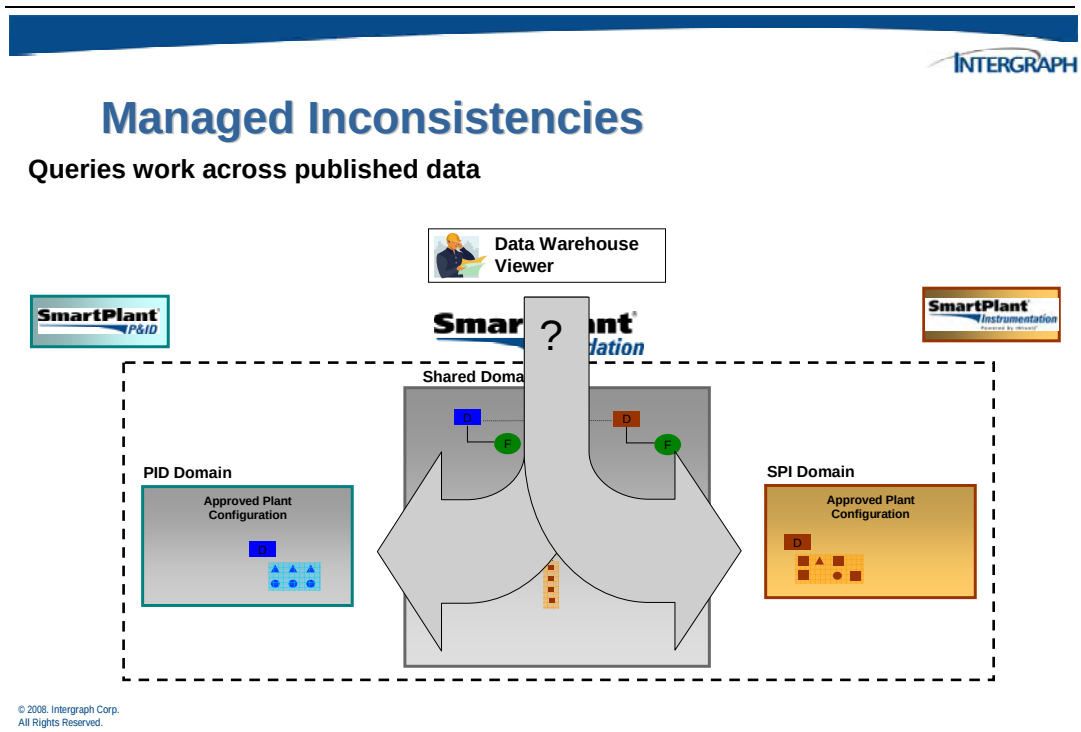


After the data contained in the associated files is loaded into the necessary tool domain, a proxy object per object is created in the shared object domain that point to the data in the tool domain. Where there is a common object used by multiple tools, a **SameAs** relationship will be issued, and rather than a new shared object being created, a shared object that already exists will be linked instead.



1.6.1 Managing Data Inconsistencies

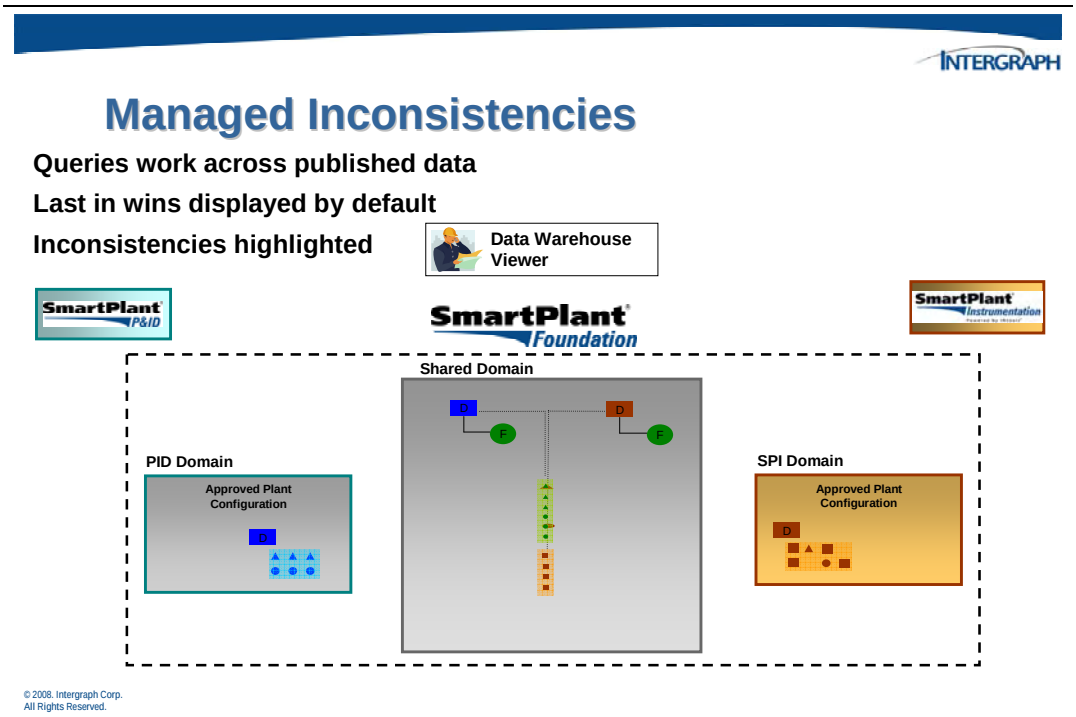
The general rule for navigating data in the data warehouse is that the user always starts any activity from the core shared object. To find an item in the data warehouse, the standard SmartPlant Foundation ***Quick Find*** functions are configured on the shared object interfaces.



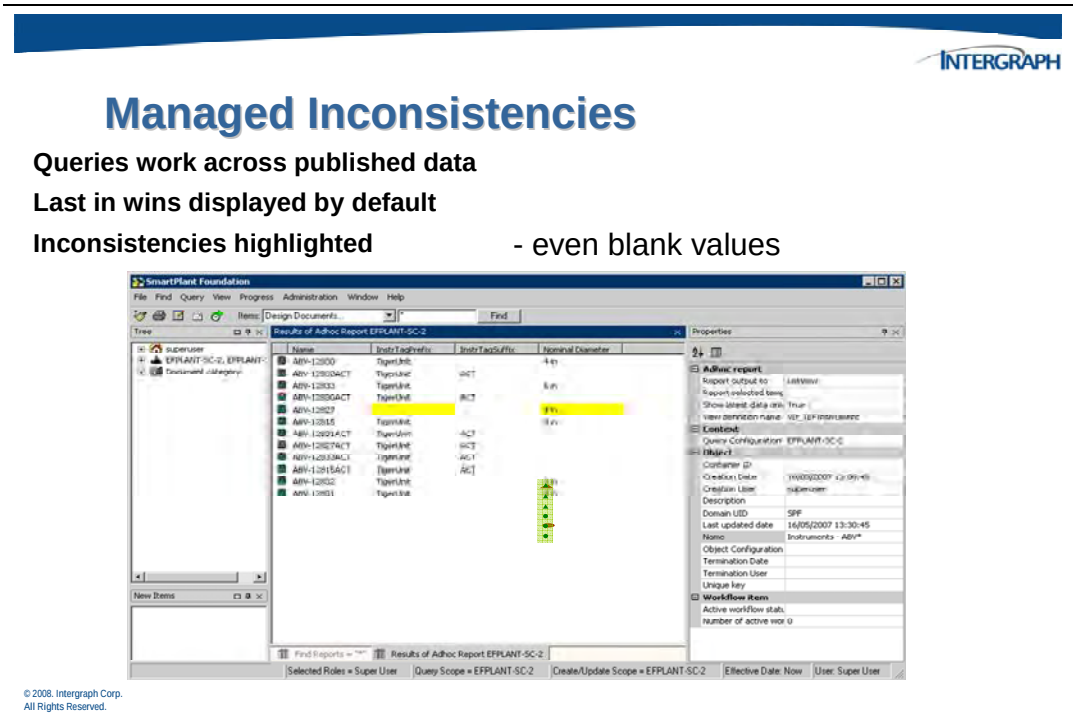
The standard detail form functionality is used to navigate from the current shared object to the associated “comprised” tool object. SmartPlant Foundation then searches for the most specific view definition that it can find and then uses this view (along with the associated graph definition) to gather up and display the data.

Although the user is unable to immediately display the different values published, the fact that the property is a different color highlights an inconsistency. If the different values need to be viewed, then using either the view inconsistencies form or the history form all values can be viewed.

If any shared data is found, then the latest value is used in the display.



Where a shared property has been published with different values from different tools, the display highlights (by displaying a different background color) the value. Although the user is unable to immediately display the different values published, the fact that the property is a different color highlights an inconsistency.



From the shared object, a new navigation option displays the results from each domain in a table format.

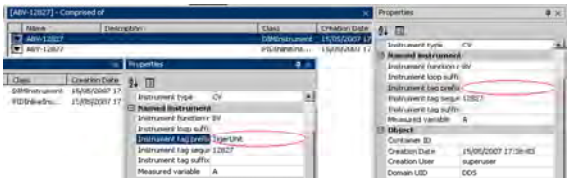
Managed Inconsistencies

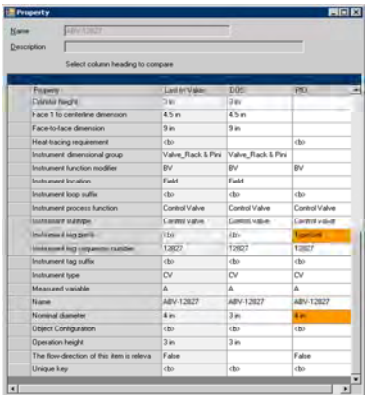
Queries work across published data

Last in wins displayed by default

Inconsistencies highlighted - even blank values

View constituent data and inconsistencies





© 2008. Intergraph Corp.
All Rights Reserved.

Domain Summary

Domains separate the data published by the tools

Domains separate the working and published data in SPF

The data warehouse displays last in wins by default

Inconsistencies are highlighted

View constituent data as published

View inconsistencies

Provides future for support of status and ownership

© 2008. Intergraph Corp.
All Rights Reserved.

CHAPTER 2

SmartPlant Foundation Loader and Schema Import Wizard

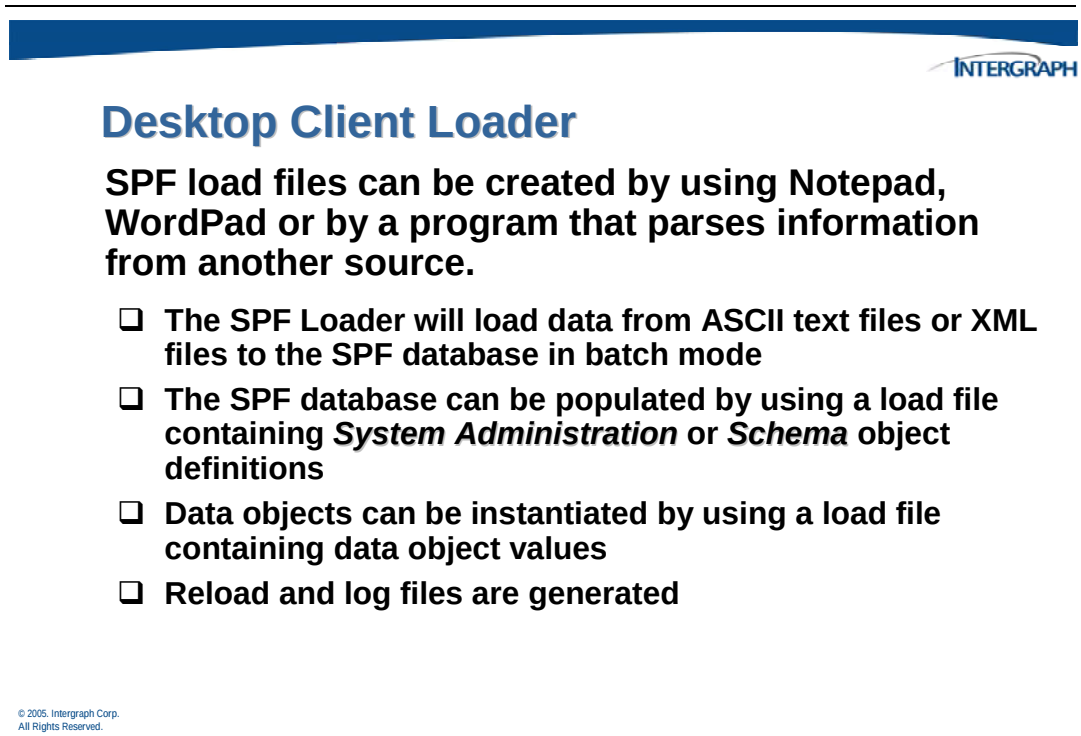
2. Understanding the Loader Interface

The SmartPlant Foundation Loader software allows you to use delimited text files to import data, relationships, and configuration information into the SmartPlant Foundation database.

The SmartPlant Foundation Loader connects with the data warehouse, administers the load, applies lifecycle information, performs error handling, and creates reload files.

The SPF Loader allows a system administrator to:

- load object data into the database using an ASCII text file or an XML file as input
- load administrative objects into the database using a specially formatted file containing system configuration definitions



You can use the **Export** commands in the *Desktop Client* to generate load files from the existing SmartPlant Foundation database.

Desktop Client Loader

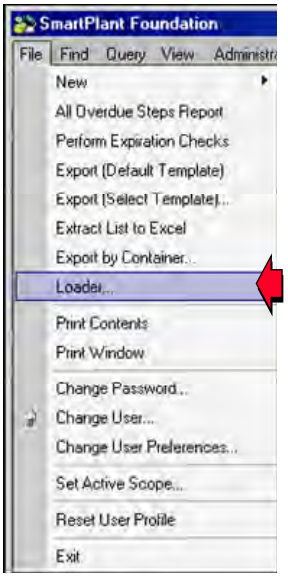
- The SmartPlant Foundation Loader is accessible from within the Desktop Client.
- The SmartPlant Foundation Loader will process text files (NLF format), data warehouse files (dwh), and XML files.
- You can build lists of load files to be run consecutively by the system. Features are provided to modify the order in which the files will be processed.
- After processing a load file through the loader, the system can display the new objects in the *New Items* window in the Desktop Client.

© 2005. Intergraph Corp.
All Rights Reserved.

The SPF Loader is started from the Desktop Client *File* menu.

Using Smart Plant Foundation Loader

- From the menu bar, click ***File > Loader***.

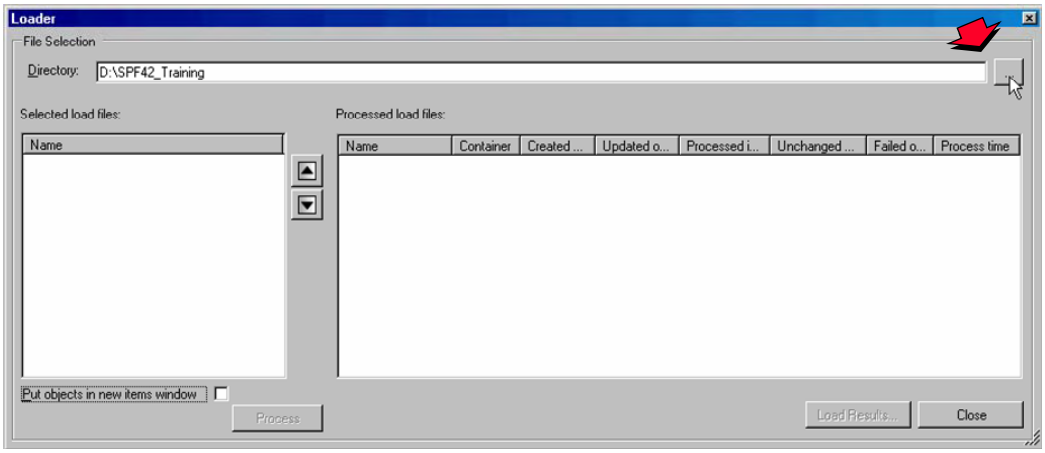


© 2005. Intergraph Corp.
All Rights Reserved.

To find the file or files you want to process with the loader, click the browse button at the top of the window.

Using Desktop Client Loader

- **Use the browse button to find the location of the loader files.**



© 2005. Intergraph Corp.
All Rights Reserved.

The following describes the fields on the *Loader* dialog:

- **Directory** - Allows you to select the files for processing. Click to browse to the files to load.
- **Selected Load files** - Lists the names of the files to be loaded.
- **Move Selected Item Up** - Moves the selected file up one position.
- **Move Selected Item Down** - Moves the selected file down one position.

Process Load Files

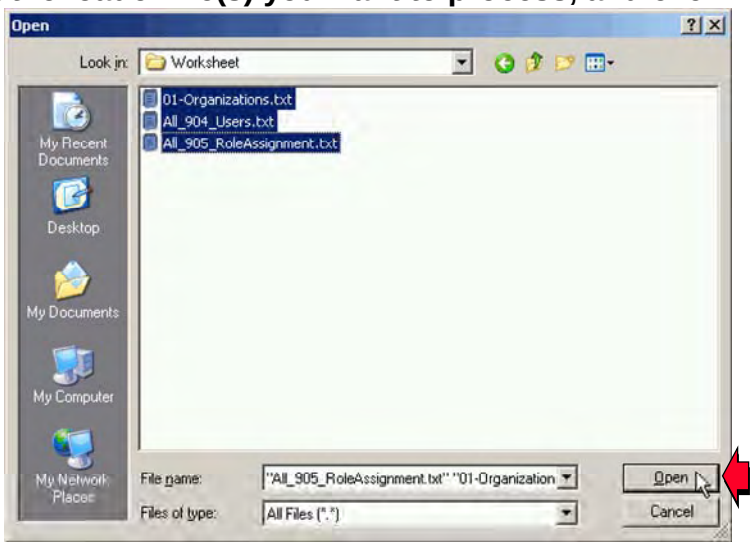
- **Name** - Displays the name of the load file
- **Container** - Displays the container to which objects are related.
- **Created objects** - Displays the number of objects created.
- **Updated objects** - Displays the number of objects updated
- **Processed instructions** - Displays the number of process instructions executed.
- **Unchanged objects** - Displays the number of objects that are unchanged.
- **Failed objects** - Displays the number of objects that failed to load
- **Process Time** - Displays the amount of time to load the file.

- **Put objects in new items window** - Displays new objects in the New Items window.
- **Process** - Processes all files in the Selected load files list.
- **Load Results** - Displays the results of the processed load files in the Load Results tree view.

Using the standard Microsoft ***Open*** dialog box, find one or more files to load.

Using Desktop Client Loader

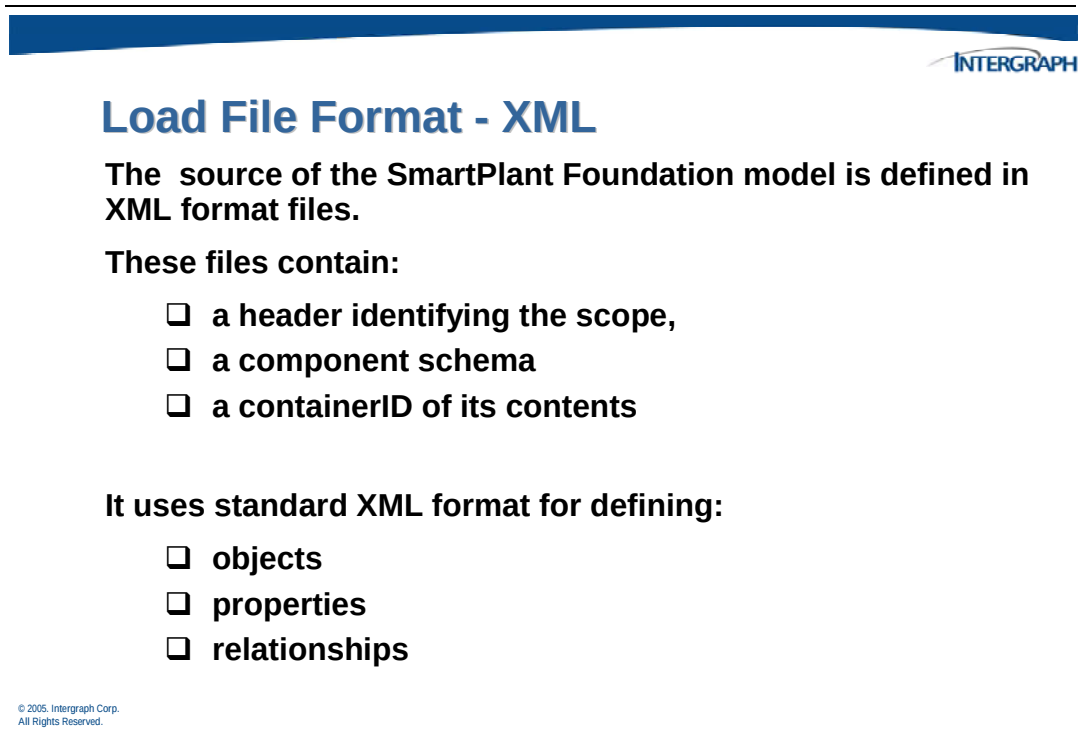
- **Select the loader file(s) you want to process, and click *Open*.**



© 2005. Intergraph Corp.
All Rights Reserved.

2.1 XML Load Format

When creating load files, an alternative to using the NLF format is XML. It is recommended that the NLF format be used for administrative or data objects and the XML format be used for schema objects. Schema XML files can be easily created using the Schema Editor.



An example of an XML load file is shown below:

```
<Container CompSchema="SPFReferenceComponent" Scope="Data" ContainerID="Plant.Admin">
   <FunctionalUnit>
<IObject UID="UNT_PlantA_U101" Name="U101" Description="Unit 101" />
<IFunctionalUnit />
<IPBSItemCollection />
<IPBSItem />
<IPlannedFacility />
  </FunctionalUnit>
  <Rel>
<IObject UID="UNT_PlantA_U101-DDCM_PlantA_C056-KV-P-XC-2002-01" />
<IRel DefUID="FunctionalUnitDeliverable" UID2="DDCM_PlantA_C056-KV-P-XC-2002-01-42"
      OrderValue="0" UID1="UNT_PlantA_U101" IsRequired="False" />
  </Rel>
</Container>
```

SmartPlant Foundation Loader can operate in three modes: **Create**, **Update** and **Merge**. The mode can be set once per load file, and should be the first active statement in the load file. The following examples shown use an *XML* style format.

Load File Mode - XML

- Specifying Type='Create' in the container means that only new objects will only be created. If they exist an error is raised and the transaction rolled back:

  <Container CompSchema='SPFComponent' config='SPF42' Type = 'Create'>

- Specifying Type='Update' in the container means that this container can only update existing objects. If any objects to be updated are not found an error is raised and the transaction rolled back:

  <Container CompSchema='SPFComponent' config='SPF42' Type = 'Update'>

© 2005. Intergraph Corp.
All Rights Reserved.

Load File Mode - XML

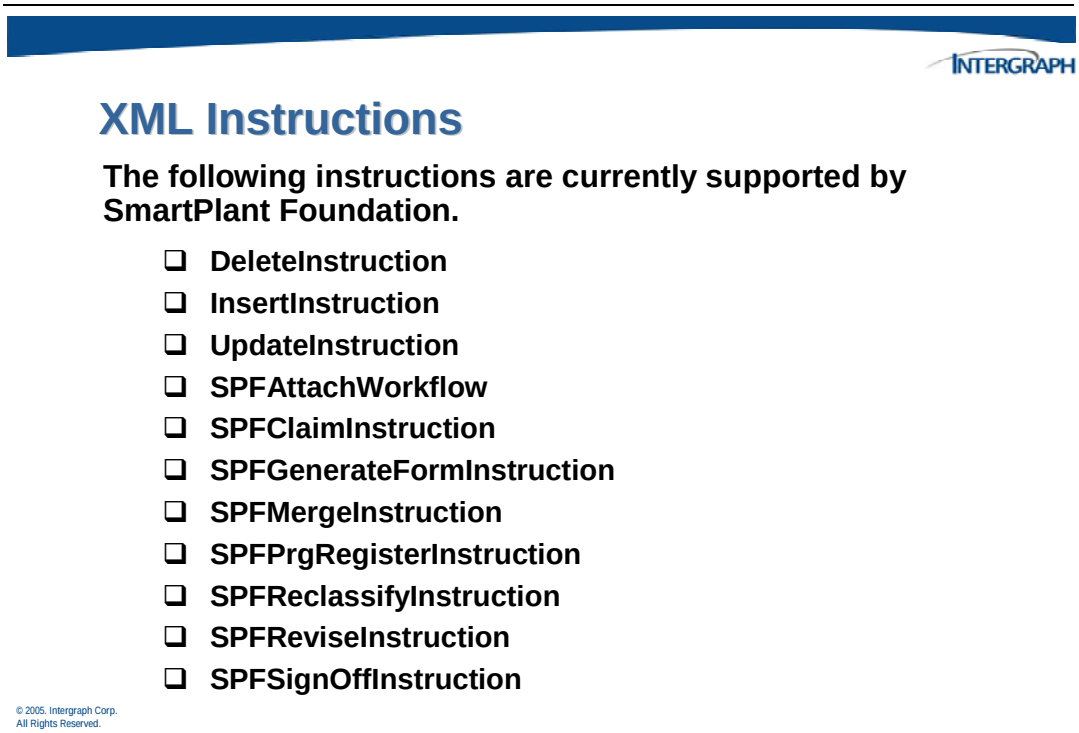
- If you don't specify a type , the 'Merge' mode is assumed. That is, If an object exists - update it , if it doesn't – create it:

  <Container CompSchema='SPFComponent' config='SPF42' >

© 2005. Intergraph Corp.
All Rights Reserved.

2.1.1 XML Loader Instructions

In addition to being able to create and update object and relationship data the xml can be constructed to perform certain SmartPlant Foundation instructions.



An example of each instruction is shown below:

Delete Instruction

```
<DeleteInstruction>
  <IObject UID="{f3edefd3-3a00-4360-a434-b19c4a825d5a}" Name="Delete_SPFAuthSignature" />
  <IRefObject RefClass="PropertyDef" RefUID="SPFAuthSignature" RefName="SPFAuthSignature"
    />
  <IDeleteInstruction DeleteTransition="Deleted" />
  <IInstruction />
</DeleteInstruction>
```

A Delete instruction can also be used for terminate by substituting the DeleteTransition parameter with the value Terminated.

```
<DeleteInstruction>
  <IObject UID="{f3edefd3-3a00-4360-a434-b19c4a825d5a}" Name="Delete_SPFAuthSignature" />
  <IRefObject RefClass="PropertyDef" RefUID="SPFAuthSignature" RefName="SPFAuthSignature"
    />
  <IDeleteInstruction DeleteTransition="Terminated" />
  <IInstruction />
</DeleteInstruction>
```

Insert Instruction

```
<InsertInstruction>
  <IObject UID="{349c5957-e5a5-42b1-959f-8313155ca0dc}" Name="Insert_ISPFFTRItem" />
  <IRefObject RefClass="InterfaceDef" RefUID="ISPFFTRItem" RefName="ISPFFTRItem" />
  <IPropValues
        PropValues="IObject\%UID\%ISPFFTRItem\%IObject\%Name\%ISPFFTRItem\%IObject\%
    Description\%Interface to support FTR\%ISchemaObj\%DisplayName\%ISPFFTRItem"
    AddedInterfaces="IObject~IInterfaceDef~IInterfaceRole~IPropCategory~IEnumMetadata~ISchem
    aObj~IInterfaceDefExt" />
  <IInsertInstruction />
  <IInstruction />
</InsertInstruction>
```

Update Instruction

```
<UpdateInstruction>
  <IObject UID="Update_UID" Name="Update_UID" />
  <IRefObject RefClass="PropertyDef" RefUID="UID" RefName="UID" />
  <IPropValues PropValues="IPropertyDefExt\%HistoryNotRetained\%True"
AddedInterfaces="IPropertyDefExt" />
  <IInstruction />
  <IUpdateInstruction />
</UpdateInstruction>
```

Progress Register Instruction

```
<SPFPrgRegisterInstruction>
  <IObject UID="Delete_ISPFAdhocReport.MTH_CopyIfOwner"
        Name="Delete_ISPFAdhocReport.MTH_CopyIfOwner" />
  <IRefObject RefClass="Rel"
RefUID="ISPFAdhocReport.MTH_CopyIfOwner"
    RefName="ISPFAdhocReport.MTH_CopyIfOwner" />
  <ISPFPrgRegisterInstruction
SPFPrgInstructionTimestring="T1"
SPFPrgInstructionPlannedDate="20/04/2007"/>
  <IInstruction />
</SPFPrgRegisterInstruction>
```

Revise Instruction

```
<SPFReviseInstruction>
  <IObject UID="Revise_001" Name="Revise_001" />
  <IRefObject RefClass="SPFDesignDocRevision"
RefUID="{68Y69856458GGKH}" RefName="DocABC" />
  <ISPFReviseInstruction SPFReviseInstRevScheme="01A"
SPFReviseInstMajorRevision="01"
SPFReviseInstMinorRevision="B" etc ..../>
  <IInstruction />
</SPFReviseInstruction>
```

SignOff Instruction

```
<SPFSignOffInstruction>
  <IObject UID="SignOff_001" Name="SignOff_001" />
  <IRefObject RefClass="SPFDesignDocRevision"
RefUID="{68Y69856458GGKH}" RefName="DocABC" />
<ISPFSignOffInstruction
SignOffInstructionComment="All is OK"
SignOffInstructionAllowWorkingRefs="True" />
```

```
  <IInstruction />
</SPFSignOffInstruction>
```

Reclassify Instruction

```
<SPFReclassifyInstruction>
  <IObject UID="Reclassify_FLD-001" Name="Reclassify_FLD-001" />
  <IRefObject RefClass="SPFFolder"
RefUID="FLD-001"
RefName="ISPFAdhocReport.MTH_CopyIfOwner" />
  <ISPFReclassifyInstruction SPFNewClassificationUID="MyFolder" />
  <IInstruction />
</SPFReclassifyInstruction>
```

Generate Form Instruction

This instruction is used to generate a default form definition for the class definition or relationship definition named or all class definitions associated with a specific component schema

```
  <SPFGenerateFormInstruction>
    <IObject UID="Instruction_GenForm-PBSAuthoringComponent"
      Name="GenForm-PBSAuthoringComponent" />
    <IRefObject RefClass="CompSchema" RefUID="PBSAuthoringComponent" />
    <IInstruction />
    <ISPFGenerateFormInstruction />
  </SPFGenerateFormInstruction>
```

Claim Instruction

```
<SPFClaimInstruction>
  <IObject UID="Claim_Tag001" Name="Claim_Tag001" />
  <IRefObject RefClass="DEVTag" RefUID="{4687889FDT98760L}" RefName="Tag001" />
  <ISPFClaimInstruction SPFClaimInstructionComment="External object claim" />
  <IInstruction />
</SPFClaimInstruction>
```

Attach Workflow Instruction

```
<SPFAttachWorkflowInstruction>
  <IObject UID="AWF_Tag001" Name="AWF_Tag001" />
  <IRefObject RefClass="DEVTag" RefUID="{4687889FDT98760L}" RefName="Tag001" />
  <ISPFAttachWorkflowInstruction SPFWorkflowTemplateUID="TagWorkflow001" />
  <IInstruction />
</SPFAttachWorkflowInstruction>
```

Merge Instruction

```
<SPFMergeInstruction>
  <IObject UID="Merge_Tag001" Name="Merge_Tag001" />
  <IRefObject RefClass="DEVTag" RefUID="{4687889FDT98760L}" RefName="Tag001" />
/S  <ISPFMergeInstruction SPFMergeInstructionComment="External object claim" />
  <IInstruction />
<SPFMergeInstruction>
```

2.2 NLF Load File Format

A load file uses delimited text to define object data and their relationships for load into the SmartPlant Foundation system. The syntax contained in a load file is used to specify information about each type of object. Some of the examples shown use a *Native Language Format* (NLF) style format.

Just like for XML format files, the SmartPlant Foundation Loader can operate in three modes for NLF: **Create**, **Update** and **Merge**. The mode can be set once per load file, and should be the first active statement in the load file.

INTERGRAPH

Load File Mode - NLF

- **Create mode is used only to create new objects. If any data collisions are detected and an error occurs, you are informed that the object is not loaded. The following syntax is used to set the Create mode for a load file:**

  Method|Create

- **Update mode is used to update existing objects. If any objects to be updated are not found and an error occurs, you are informed that the object is not loaded. The following syntax is used to set the Update mode for a loadfile:**

  Method|Update

© 2005. Intergraph Corp.
All Rights Reserved.

Load File Mode - NLF

Load File Modes (con't):

- **Merge mode updates an existing object or creates the object if it does not already exist. Use the following syntax to set the Merge mode for a load file:**

  Method|Merge

NOTE: Merge mode is the default mode for SmartPlant Foundation Loader. If you do not specify a Method| directive, the load file will be imported in Merge mode.

© 2005. Intergraph Corp.
All Rights Reserved.

Each line of a NLF file contains information about the objects to be loaded into SmartPlant Foundation.

Load File Format

- **File uses NLF format (Native Language File)**
- **A dot prefix (.) defines a new object**
- **Two dots (..) indicate the information belongs to (or is associated with) the key object**
- **A pipe (|) separates the data fields**

© 2005. Intergraph Corp.
All Rights Reserved.

The Loader still supports the NLF format. The reason for this is to support both backwards compatibility and to support an external load format where UIDs are unknown. Load files constructed this way will be sent to the server and parsed through a load pre processor. During this step the UIDs of the object are determined and Xml is constructed. The Xml is then submitted to the same methods that the Xml loader described above is submitted to, those making sure that all load operations parse through the same code. Because the NLF format does require this extra step to parse the data then the operation is slightly longer than submitting Xml directly.

The following describes the format for an NLF load file.

Object Type Format - NLF

Format	Description	Example
.Class_Type	Defines the classdef of the object .	. DEVTag
..Name|Value	Defines the name of the object instance in the database.	..Name| CW501
..UID|Value	Defines the UID of the object instance in the database.	..UID|UID-U101
..Description|Value	Defines the description of the business object instance in the database.	..Description|NLF Tag Create
..PropertDef|value|UOM|Type	Defines the value of the given property name. If the property is of type UoMListType, the third field includes the name of UOM.	..DEVOverallLength|10|m

© 2005. Intergraph Corp.
All Rights Reserved.

Object Type Format - NLF

Format	Description	Example
..ClassDef|Value|Reldef Name	Defines the internal name of the related object such as Area, Discipline, or System. Argument 1 is used to define the name of the object and argument 2 names the edge relationship definition. Note! A relationship edge is the name of the relationship def with either "_12" or "_21" appended to the name depending on the direction from the defined object.	..SPFTemplateRevision|DocABCD|DEVDocumentTag_21
..InterfaceDef|Value	Allows an optional interface to be added to the object	..IDEVEquip

© 2005. Intergraph Corp.
All Rights Reserved.

The load file for creating tag *CW501* in NLF Format and relating it to a document *DocABC* is shown below:

.DEVTag

..Name|CW501

..Description|NLF Tag Create

..DEVTagCategory|Process equipment

..DEVTagType|Process vessel

..DEVOverallHeight|900|mm

..DEVOverallWidth|3.2|m

..DEVOverallLength|4.1|m

..SPFTemplateRevision|DocABC|DEVDocumentTag_21

2.2.1 NLF Instructions

NLF also supports a subset of SmartPlant Foundation instructions. These can be triggered by defining the object that exists in the database followed by the keyword **Action** and then one of the defined Actions as listed below.

NLF Instructions

Format	Description
..Action|Delete	If set will generate a DeleteInstruction for the named object and set the argument DeleteTransition to Delete. The result will delete the object, its relationships and any associated object that is defined using the delete_12 and delete_21 flags on the reldef
..Action| ProgressRegister| TimeString|PlannedDate	If set will generate a SPFPrgRegisterInstruction and submited arg1 as the SPFPrgInstructionTimestring and arg2 as SPFPrgInstructionPlannedDate. The date format is dd/mm/yyyy. The instruction will register the name object and associate it with a workpack.
..Action|Revise|RevScheme| MajorRevision|MinorRevision	This action works on Document objects only. If set will generate a SPFReviseInstruction and submited arg1 as the SPFReviseInstRevScheme, arg2 as SPFReviseInstMajorRevision and arg3 as SPFReviseInstMinorRevision.. The instruction will validate that the document can be revised before performing the revise function.

© 2005. Intergraph Corp.
All Rights Reserved.

NLF Instructions

Format	Description
..Action|SignOff|	**This action works on Document objects only. If set will generate a SPFSignOffInstruction and submited arg1 as the SignOffInstructionComment, and arg2 as SignOffInstructionAllowWorkingRefs. The instruction will validate that the document can be signedoff before performing the signoff function.**
..Action|Terminate	**If set will generate a DeleteInstruction for the named object and set the argument DeleteTransition to Terminated. The result will terminate the object, its relationships and any associated object that is defined using the delete_12 and delete_21 flags on the reldef**
..Action|Reclassify| **New class definition**	**If set will generate a SPFReclassifyInstruction for the named object. The new class definition supplied as arg 1 will be checked to see if it belongs in the same shared object group as the existing class definition of the object. If it does reclassification takes place.**

© 2005. Intergraph Corp.
All Rights Reserved.

Format	Description
..Action|Claim|ClaimComment	If set will generate a SPFClaimInstruction for the named object and set the SPFClaimInstructionComment to the value specified by arg1. The result will claim the object, its relationships and any associated object that is defined using the claim_12 and claim_21 flags on the reldef
..Action|AttachWorkflow|Workflow UID	This action will generate a SPFAttachWorkflowInstruction for the named object and set the Workflow UID to the value specified by arg1. The result will attach the name workflow to the object if it is valid to do so.
..Action|Merge|Comments	This action will generate a SPFMergeInstruction for the named object and set the SPFMergeComment to the value specified by arg1.
..Action|CreateNewVersionIfExists	This action is specific for files and will auto create the next version of document if there is one already

An example NLF file with an instruction is shown below:

.DEVTag

..Name|CW506

..Action|Terminate

2.2.2 Field Types for NLF

If the first field of the data is neither a keyword (Name, Description, or Classification) nor is it found in the list of related objects (System, Area, Discipline), then the Loader assumes the field is a property.

Unlike previous versions (3.x), there is no longer a requirement to append the property type as input as 4x will determine the type from the property definition held within the database.

Date Time Handling

The Loader supports 2 types of date/time formats: Server and Local.

- Server format is: yyyy/mm/dd-hh:mm:ss:mmm. The Server format supports this format whether the time is specified or not. If a time is specified, then the time is adjusted using regional time settings. If no time is specified, then the string 00:00:00 (indicating midnight local time) is appended. This time string is then converted to GMT time using regional settings.
- Local format is different in that the date is entered based on the local date format specified in your Control Panel Settings. For example, May 31, 2004 could be entered as 05/31/2004 or 31/05/2004 depending on your location. The local time format is: hh:mm:ss and does not support milliseconds. If a date and time are specified, the "-" separator is used as shown in the Server format. If no time is specified, then the local date is converted to the server format and the time string 00:00:00 (indicating midnight local time) is appended. This time string is then converted to GMT time using regional settings. All conversions are done before sending the date and time to the server.

CreateConfig Setting

It is possible to override the Create Configuration that is set in the client when loading load files. If this is not set then the data is loaded using the clients' current Create configuration. Configuration independent classes will ignore this setting.

Example:

.SetConfiguration

..Configuration|SPF42

ContainerID

The containerID tag allows all objects that are created to be stamped with the value defined in argument 1

Example:

ContainerID|MyLoadedData

2.3 Loading Documents Using NLF

In order to make it easier for NLF to load documents a user can either fully define the document by specifying each class and all relationships that make up a document or they can define the document in a short hand format where only the document master in full is defined but this has special keywords on it that makes it easier to create the revision and version objects.

If a revision or version property or the keyword Action is detected on the document master class instance then the revision, version and the associated rels will be referenced or generated automatically.

The way in which this is achieved is by first finding out the associated revision and version class definition. This is done by expanding up the SPFIncludes relationship from the specified document master class definition.

The name and description of the revision and version objects will be the same as the document master.

The following is an example of a document load file.

INTERGRAPH

SPF Load File Example - NLF

```
# =========Create Design Document ===================
.BeginContainer

.SetConfiguration
..Configuration|SPF42

.SPFDesignDocMaster
..Name|PM-2000-CD-0043
..Description|Equipment Data Sheet - Centrifugal Pump
..SPFDataAccessGroup|ENGINEER|SPFItemOwningGroup_12
..SPFDocumentClass|Civil Details|SPFPrimaryClassification_21
..SPFTitle|PM-2000-CD-0043
```

© 2005, Intergraph Corp.
All Rights Reserved.

SPF Load File Example - NLF

```
..SPFRevisionScheme|RevA1|SPFDocRevisionRevisionScheme_12
..SPFMajorRevision|A
..SPFMinorRevision|1
..SPFDocState|RESERVED
..SPFRevState|WORKING
..SPFDocVersion|1
```

Note: EnumList properties must match the values from the corresponding enumerated list.

© 2005. Intergraph Corp.
All Rights Reserved.

2.3.1 Relationship Data

In order to load relationship data into SmartPlant Foundation, the reldef edge name is used as the *Relation Type* when defining the object type but. Relationship data uses specific keywords to define each object.

There is sometimes the need to generate relationships that relate other relationships to objects such as in the case of conditions that are associated with the relationship that goes between a class definition and an access group.

Example NLF definition with a relationship relation defined

.SPFClassDefAccessGroup

..Name1|DEVTag

..Name2|VIEWONLY

..SPFCondition|IsConfigMerged|SPFClassDefAccessGroupCondition_12

The following are some load examples for a plant, area and unit with relationships.

INTERGRAPH

SPF Load File Example - NLF

Relationship example -

 # ======= PLANT =======

.SPFPlant

..Name|Madison

..Description|New Plant for PBS

..SPFConfigurationStatus|Active|SPF ConfigurationConfigurationStatus_12

..ContainerID|Trn.Mods

.SPFConfigurationTree

..Name1|ConfigurationTop

..Name2|Madison

..ContainerID|Trn.Mods

 # ======= AREA =======

.BeginContainer

.SetConfiguration

..Configuration|Madison

.SPFFunctionalArea

..Name|Area1

..Description|New Area for Madison Plant

..SPFPlant|Madison|SPFPlantAreas _12

..UniqueKey|AR_Madison_Area1

..ContainerID|Trn.Mods

.EndContainer

© 2005. Intergraph Corp.
All Rights Reserved.

SPF Load File Example - NLF

Relationship example (con't) -

 # ======= UNIT =======

.BeginContainer

..ContainerID|Trn.Mods

.EndContainer

.SetConfiguration

..Configuration|Madison

.SPFFunctionalUnit

..Name|Unit1

..Description|New Unit for Area 1

..SPFFunctionalArea|Area1|SPFFunctional AreaUnits_12

© 2005. Intergraph Corp.
All Rights Reserved.

Relationship Definition Keywords

Format	Description	Example
.RelDef Name	Defines the name of the reldef	. DEVDocumentTag the direction of the relationship should always be specified.
..Name1|value	Defines the name of the left (first) object. There is a special case for relationship relations that is defined below	..Name1| DocABC
..Name2|value	Defines the name of the right (second) object.	..Name2| CW503
..<propertyName>/Value	Define any other property that belongs on the relationship	..OrderValue|9
..UniqueKey1|Value	As an alternative to Name1 the uniquekey can be specified	..UniqueKey1|DM_DocABC_A_1_1
..UniqueKey2|Value	As an alternative to Name2 the uniquekey	..UniqueKey2|TG_CW503

Format	Description	Example
	can be specified	
UID1|Value	As an alternative to Name1 the UID1 can be specified	..UID1|{4fytj1664x56)
UID2|Value	As an alternative to Name2 the UID2 can be specified	..UID2|{45516dyyt66)
..Action|Value|Arg1	See previous section for detail	..Action|Terminate
..<interface name>	Defines a optional interface that can be added onto the relationship	..SPFClaimableItem
..<Classdef>|Name|reldef	Defines a relationship relation. If the cardinality is set to max =1 and the objet is being updated then the existing rel is terminated.	..SPFCondition|IsConfigMerged|SPFClassDefAccessGroupCondition_12

2.3.2 Loading a Document with an Attached File

A load file can be created to attach a file to an object (such as a document) within SmartPlant Foundation and transfer it to the SmartPlant Foundation vault. To upload and attach a file, two NLF statements are required. The first is an *UploadInstruction* to move the physical file from the local host to a server and the second is the file object itself, which represents the meta-data that is created in the database.

```
.UploadInstruction
..RefClass|SPFAuthAdapterFile
..RefName|ToolMapSchema_PBSFile
..RemoveLocalFileInd|False

.SPFAuthAdapterFile
..Name|ToolMapSchema_PBSFile
..SPFLocalDirectory|C:\
..SPFLocalFileName|ToolMapSchema_PBS.xml
..ISPFDBFile
```

The upload instruction needs to reference the associated file in order for the transfer to be successful. For files that end up in the database (blob) the interface ISPFDBFile needs to be referenced, else if it is a file that ends up in the vault then a vault needs to be referenced.

SPF Load File Example - NLF

Create Document / Attach File example

.BeginContainer

.SetConfiguration

..Configuration|SPF42

.SPFDesignDocMaster

..Name|PM-2000-CD-0044

..Description|Equipment Data Sheet - Tank

..SPFDataAccessGroup|ENGINEER|SPFItemOwningGroup_12

..SPFDocumentClass|Civil Details|SPFPrimaryClassification_21

..SPFTitle|PM-2000-CD-0044

..SPFRevisionScheme|RevA1|SPFDocRevisionRevisionScheme_12

..SPFMajorRevision|A

..SPFMinorRevision|1

© 2005. Intergraph Corp.
All Rights Reserved.

SPF Load File Example - NLF

Create Document / Attach File example (con't):

..SPFDocState|RESERVED

..SPFRevState|WORKING

..SPFDocVersion|1

.UploadInstruction

..RefClass|SPFDesignFile

..RefName|tank.dgn

..RemoveLocalFileInd|False

.SPFDesignFile

..Name|tank.dgn

..SPFDesignDocVersion|PM-2000-CD-0044|SPFFileComposition_12

..SPFLocalDirectory|D:\SPF42_Training\loader

..SPFLocalFileName|tank.dgn

..SPFVault|SPF42_Vault|SPFFileVault_12

..SPFEditInd|True

..SPFViewInd|True

.EndContainer

© 2005. Intergraph Corp.
All Rights Reserved.

To auto create the next version and attach the document to this instead, use the instruction *CreateNewVersionIfExists* as shown below:

.SPFTemplateFile

..Name|XLS_Tag_Data_1

..SPFLocalDirectory|C:\

..SPFLocalFileName|XLS Tag Data 2.xls

..ISPFDBFile

..SPFTemplateVersion|XLS_Tag_Data|SPFFileComposition_12

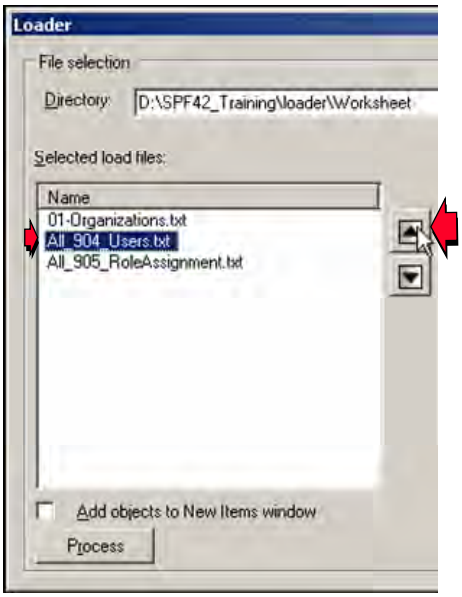
..Action|CreateNewVersionIfExists

2.4 Processing Load Files

When you are ready to begin the load process, click the ***Process*** button at the bottom of the window. Prior to selecting the Process button, organize the files in the correct order for processing.

Using Desktop Client Loader

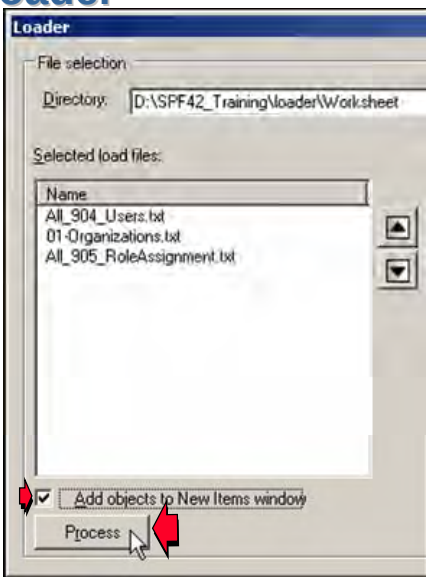
- **Highlight one of the load files and use the Up/Down arrows to select the correct order in which to process multiple files.**



© 2005. Intergraph Corp.
All Rights Reserved.

Using Desktop Client Loader

- If you want to add new items created in the database to the ***New Items*** window in the Desktop Client, check the ***Add objects to New Items window*** option.
- Click ***Process*** to start running the load file(s).

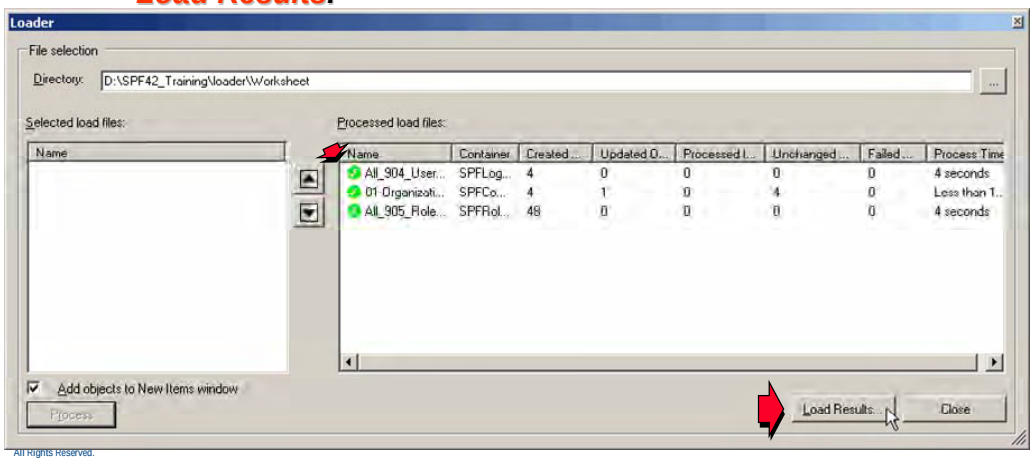


© 2005. Intergraph Corp.
All Rights Reserved.

A check box is provided to indicate that new items created with the load file should be displayed in the ***New Items*** window of the Desktop Client interface. The results of the process are displayed on the right-hand side of the window.

Using Desktop Client Loader

- Verify that the file loaded correctly. If you want to view specific details of what was done with the load file, click ***Load Results***.



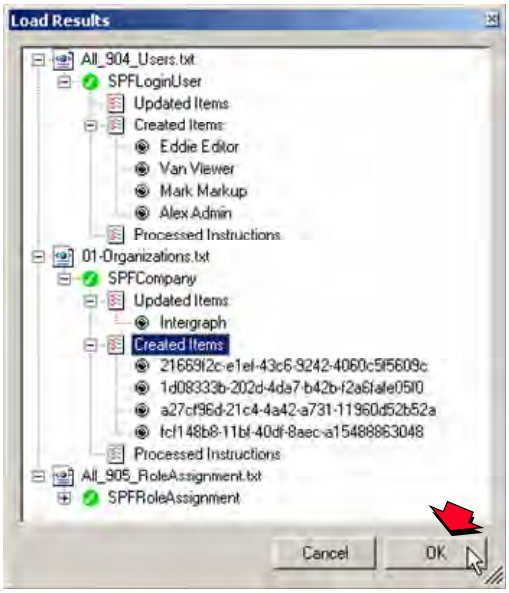
All Rights Reserved.

A green icon indicates files that were processed successfully, while a red icon indicates files that failed to load properly.

To see specific details about what was loaded when the file was processed, click the ***Load Result*** button to view more details about what the load file did.

Using Desktop Client Loader

- **When you are finishing viewing the resulting details of the load operation, click *OK*.**

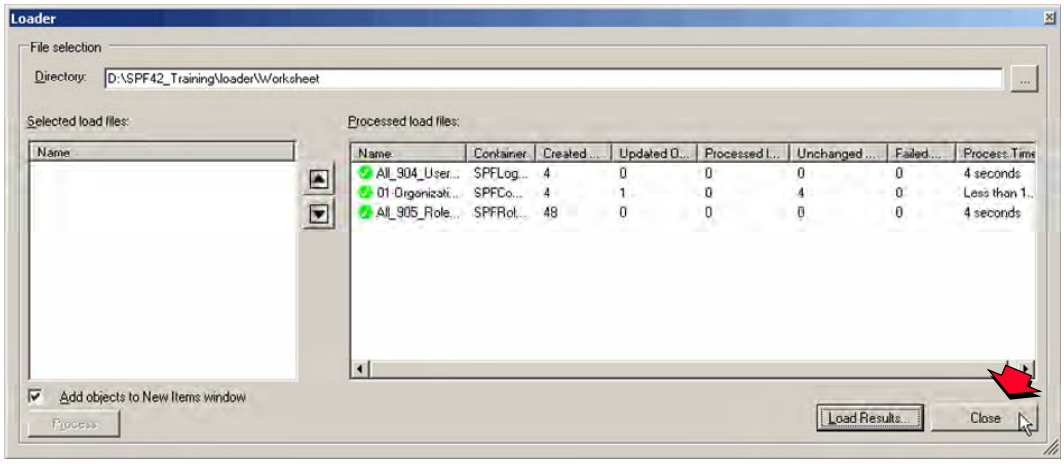


© 2005. Intergraph Corp.
All Rights Reserved.

Click the ***Close*** button to dismiss the ***Loader*** window.

Using Desktop Client Loader

- **Click *Close* to dismiss the Desktop Client Loader.**

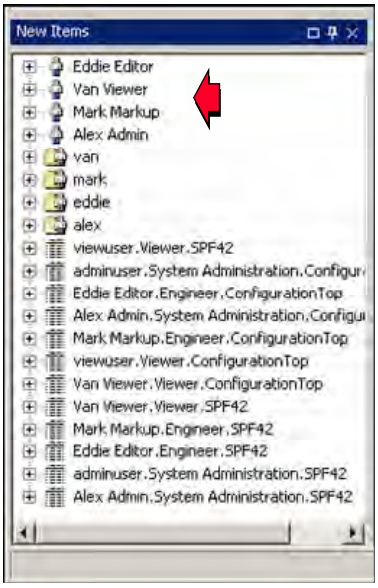


© 2005. Intergraph Corp.
All Rights Reserved.

If you enabled the ***Add objects To New Items window*** check box, new items created with the load file should now appear in that window back in the main interface.

Using Desktop Client Loader

- **Verify that the new objects appear in the *New Items* window.**



© 2005. Intergraph Corp.
All Rights Reserved.

The following examples are the load files processed above.

```
# === USERS =======

.SPFLoginUser
..SPFLoginName|alex
..SPFPassword|
..Name|Alex Admin
..Description|An administrative user
..SPFWarningDialogCount|1000
..SPFMaximumSQLLimit|2000
..SPFEmailAddress|alex@ingr.com
..SPFLastPasswordUpdate|
..SPFDaysBetweenPasswordChange|
..SPFInboxRefresh|300
..SPFNotificationType|e1EmailAndNotification
..SPFExpandToOption|New Items View
..SPFSignOffTimeOut|0
..SPFShowHiddenFile|False
..ContainerID|Trn.Mods

.SPFLoginUser
..SPFLoginName|eddie
..SPFPassword|
..Name|Eddie Editor
..Description|Ad edit/view user
..SPFWarningDialogCount|50
```

```
..SPFMaximumSQLLimit|300
..SPFEmailAddress|eddie@ingr.com
..SPFLastPasswordUpdate|
..SPFDaysBetweenPasswordChange|
..SPFInboxRefresh|300
..SPFNotificationType|e1EmailAndNotification
..SPFExpandToOption|New Items View
..SPFSignOffTimeOut|0
..SPFShowHiddenFile|False
..ContainerID|Trn.Mods

.SPFLoginUser
..SPFLoginName|mark
..SPFPassword|
..Name|Mark Markup
..Description|An edit/view user
..SPFWarningDialogCount|50
..SPFMaximumSQLLimit|300
..SPFEmailAddress|mark@ingr.com
..SPFLastPasswordUpdate|
..SPFDaysBetweenPasswordChange|
..SPFInboxRefresh|300
..SPFNotificationType|e1EmailAndNotification
..SPFExpandToOption|New Items View
..SPFSignOffTimeOut|0
..SPFShowHiddenFile|False
..ContainerID|Trn.Mods

.SPFLoginUser
..SPFLoginName|van
..SPFPassword|
..Name|Van Viewer
..Description|A view user
..SPFWarningDialogCount|50
..SPFMaximumSQLLimit|300
..SPFEmailAddress|van@ingr.com
..SPFLastPasswordUpdate|
..SPFDaysBetweenPasswordChange|
..SPFInboxRefresh|300
..SPFNotificationType|e1EmailAndNotification
..SPFExpandToOption|New Items View
..SPFSignOffTimeOut|0
..SPFShowHiddenFile|False
..ContainerID|Trn.Mods

# === COMPANY =======

.SPFCompany
..Name|Intergraph
..ISPFInternalOrg
..SPFOrgTelNumber|256-730-2000
..ContainerID|Trn.Mods


.SPFLoginUser
```

```
..Name|Alex Admin
..SPFLoginName|alex
..SPFCompany|Intergraph|SPFUserOrganization_12
..ContainerID|Trn.Mods


.SPFLoginUser
..Name|Eddie Editor
..SPFLoginName|eddie
..SPFCompany|Intergraph|SPFUserOrganization_12
..ContainerID|Trn.Mods


.SPFLoginUser
..Name|Mark Markup
..SPFLoginName|mark
..SPFCompany|Intergraph|SPFUserOrganization_12
..ContainerID|Trn.Mods


.SPFLoginUser
..Name|Van Viewer
..SPFLoginName|van
..SPFCompany|Intergraph|SPFUserOrganization_12
..ContainerID|Trn.Mods


.EndContainer

# ======= ROLE ASSIGNMENT =======

.SPFRoleAssignment
..UID|Eddie Editor.Engineer.SPF42
..Name|Eddie Editor.Engineer.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Eddie Editor|SPFRoleAssignmentUser_12
..SPFRole|Engineer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Eddie Editor.Engineer.ConfigurationTop
..Name|Eddie Editor.Engineer.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Eddie Editor|SPFRoleAssignmentUser_12
..SPFRole|Engineer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Mark Markup.Engineer.SPF42
..Name|Mark Markup.Engineer.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Mark Markup|SPFRoleAssignmentUser_12
..SPFRole|Engineer|SPFRoleAssignmentRole_12
```

```
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Mark Markup.Engineer.ConfigurationTop
..Name|Mark Markup.Engineer.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Mark Markup|SPFRoleAssignmentUser_12
..SPFRole|Engineer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Alex Admin.System Administration.SPF42
..Name|Alex Admin.System Administration.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Alex Admin|SPFRoleAssignmentUser_12
..SPFRole|System Administration|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Alex Admin.System Administration.ConfigurationTop
..Name|Alex Admin.System Administration.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Alex Admin|SPFRoleAssignmentUser_12
..SPFRole|System Administration|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|adminuser.System Administration.SPF42
..Name|adminuser.System Administration.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|adminuser|SPFRoleAssignmentUser_12
..SPFRole|System Administration|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|adminuser.System Administration.ConfigurationTop
..Name|adminuser.System Administration.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|adminuser|SPFRoleAssignmentUser_12
..SPFRole|System Administration|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Van Viewer.Viewer.SPF42
..Name|Van Viewer.Viewer.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Van Viewer|SPFRoleAssignmentUser_12
..SPFRole|Viewer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
```

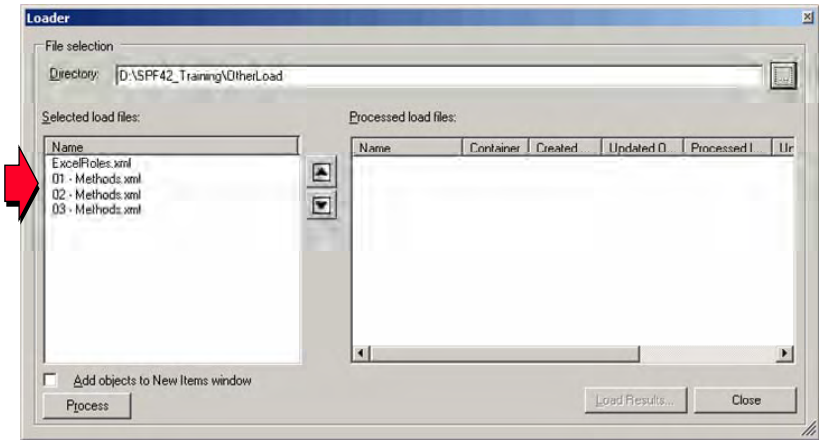
```
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|Van Viewer.Viewer.ConfigurationTop
..Name|Van Viewer.Viewer.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|Van Viewer|SPFRoleAssignmentUser_12
..SPFRole|Viewer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|viewuser.Viewer.SPF42
..Name|viewuser.Viewer.SPF42
..SPFPlant|SPF42|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|viewuser|SPFRoleAssignmentUser_12
..SPFRole|Viewer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods

.SPFRoleAssignment
..UID|viewuser.Viewer.ConfigurationTop
..Name|viewuser.Viewer.ConfigurationTop
..SPFConfigurationTop|ConfigurationTop|SPFRoleAssignmentConfigItem_12
..SPFLoginUser|viewuser|SPFRoleAssignmentUser_12
..SPFRole|Viewer|SPFRoleAssignmentRole_12
..SPFIncludeSubConfigurations|False
..ContainerID|Trn.Mods
```

2.4.1 Additional SPF Loader Features

It is also possible to run multiple load files with one command. You can build a list of files you want to load in the ***Selected load files*** window in the left-hand side of the ***Loader*** window. When you click the ***Process*** button, the utility will attempt to process the entire list.

Additional Features of the DTC Loader

- **You can build a list of load files to be processed, either by selecting multiple files on the *Open* dialog box, or by using the browse button more that once and finding additional files.**

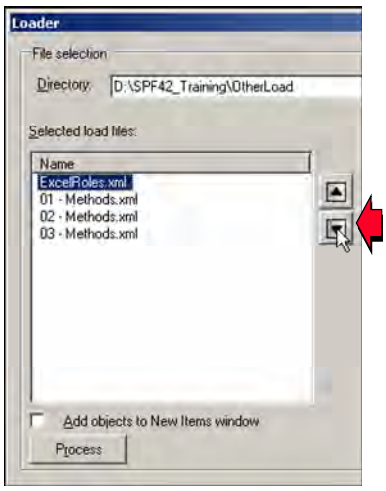


© 2005. Intergraph Corp.
All Rights Reserved.

The files in the list are processed in order from top to bottom. You can adjust this order by selecting a file and using the up and down arrows to move it in the list.

Additional Features of the DTC Loader

- **Use the up and down arrows to change the order in which the load files will be processed.**

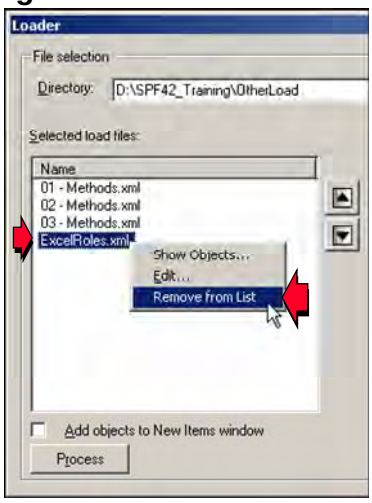


© 2005. Intergraph Corp.
All Rights Reserved.

If you change your mind about a file, you can remove it from the list of those to be processed.

Additional Features of the DTC Loader

- **To remove an item from the list of load files to be processed, right-click the file and use the *Remove from list* option.**

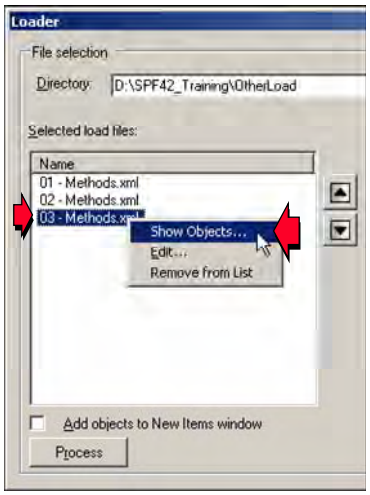


© 2005. Intergraph Corp.
All Rights Reserved.

Use the ***Show Object*** option on the shortcut menu to view the object that will be create/updated with a load file.

Additional Features of the DTC Loader

- **To view a list of the changes in a load file before processing, right-click on the file and click the *Show Objects* command.**

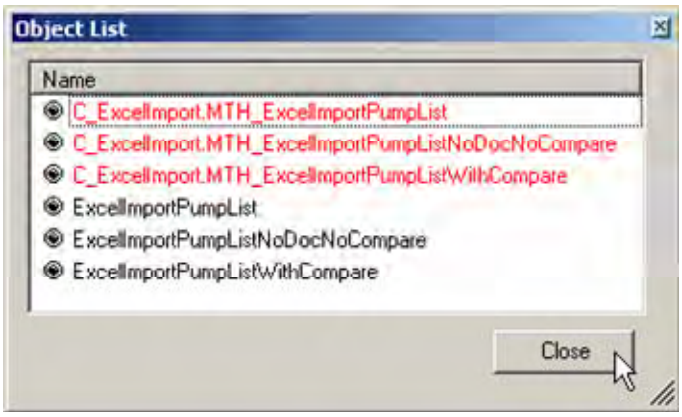


© 2005. Intergraph Corp.
All Rights Reserved.

The results appear in the ***Object List*** window.

Additional Features of the DTC Loader

- **The objects within the selected load file are then displayed in the *Object List* dialog box.**

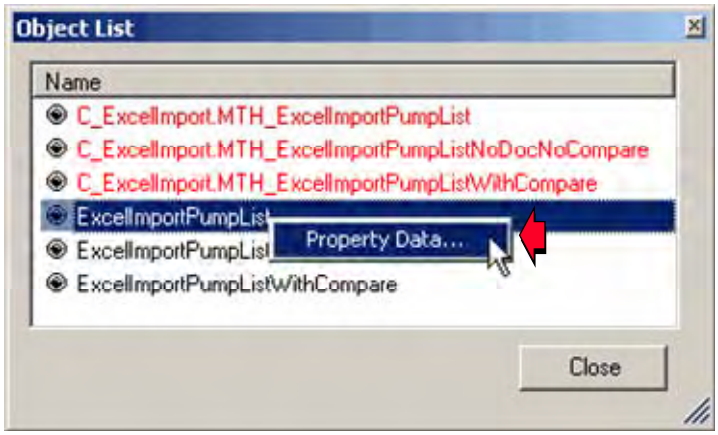


© 2005. Intergraph Corp.
All Rights Reserved.

Right-click on an object and use the ***Property Data*** command on the shortcut menu to see more information about any object in the load file.

Additional Features of the DTC Loader

- **Right-click on a specific object and click the *Property Data* option to view specific properties to be set for the object.**

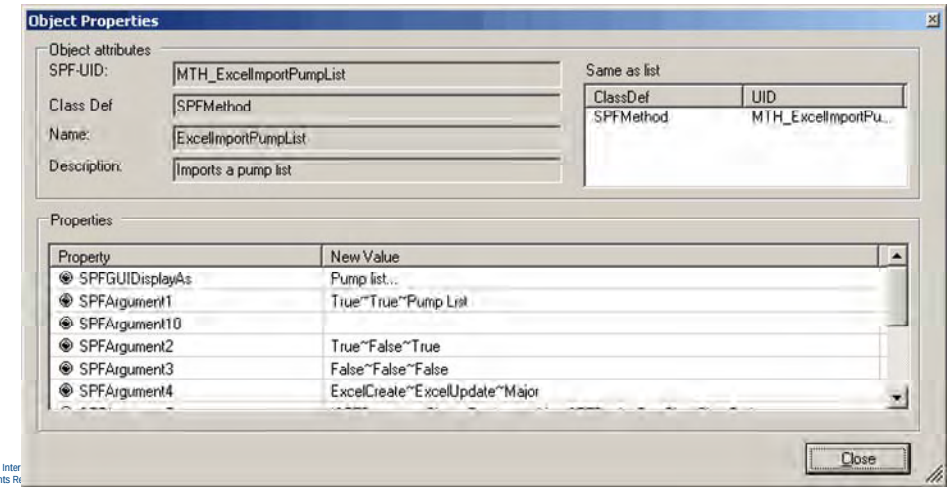


© 2005. Intergraph Corp.
All Rights Reserved.

The ***Object Properties*** dialog box show details about the object.

Additional Features of the DTC Loader

- **On the *Object Properties* dialog box, you can view the properties to be set by the load file for the specific object.**

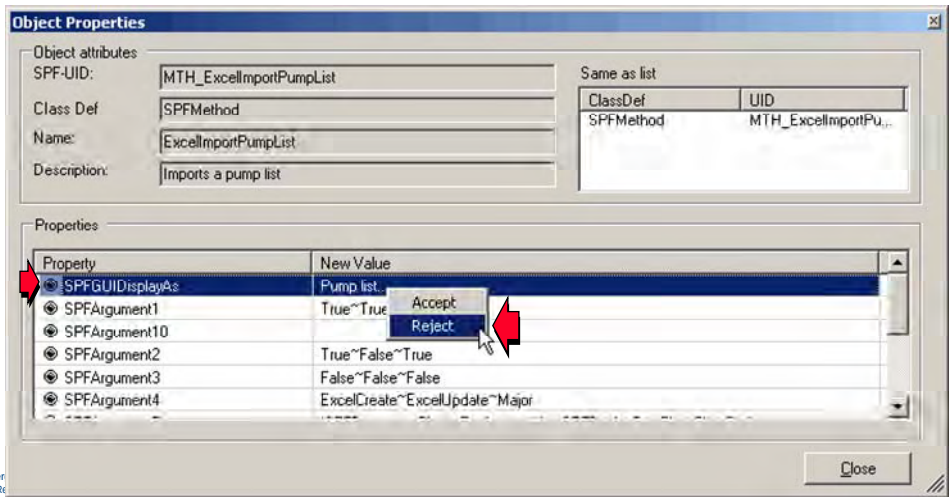


© 2005. Inter
All Rights Re

From the ***Object Properties*** dialog box, you can view specific values to be set for properties of the object. To disable the setting of any of those values, right-click on the property and use the ***Reject*** option.

Additional Features of the DTC Loader

- **Right-click on a particular property and value to choose whether to accept or reject the value being set.**

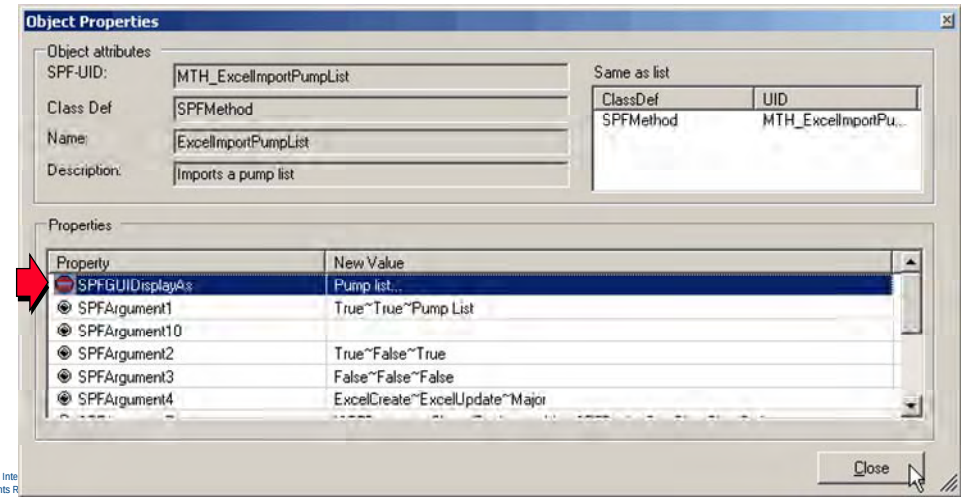


Note: You can reset a property, so that it will be processed, by right-clicking on it again and selecting the *Accept* option.

Rejected properties will not be set when the load file is processed.

Additional Features of the DTC Loader

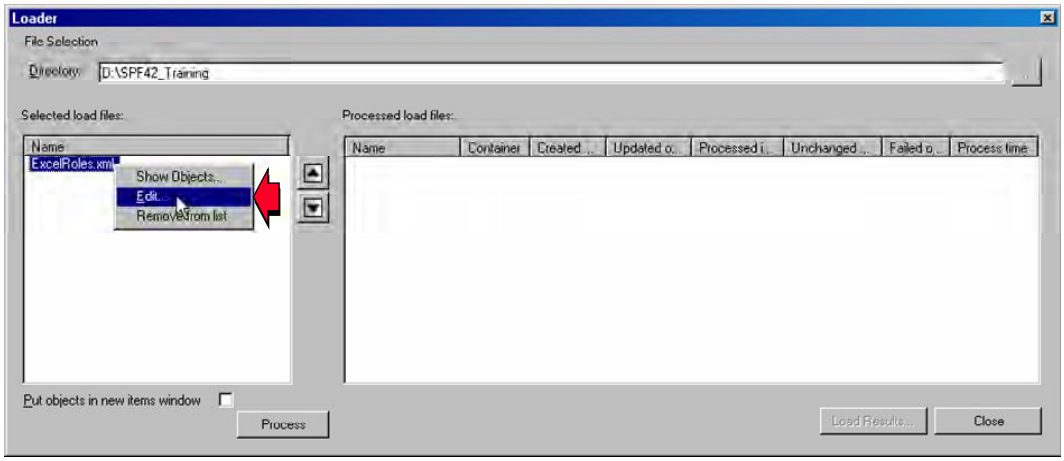
- **A rejected value will not be set when the load file is processed by the DTC Loader.**



To make a change to a load file before processing it, use the ***Edit*** option to open the file in the native application.

Additional Features of the DTC Loader

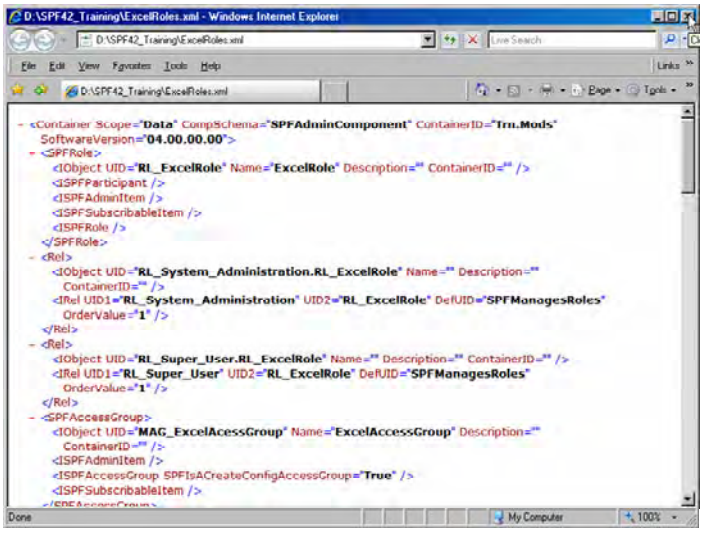
- **To make last-minute changes to a load file before processing it, right-click on the load file and click the *Edit* command.**



You can make changes to the file and save those changes before processing it with the SmartPlant Foundation Loader.

Additional Features of the DTC Loader

- **The load file is opened in its native application for editing.**



© 2005. Intergraph Corp.
All Rights Reserved.

2.5 Reload Files

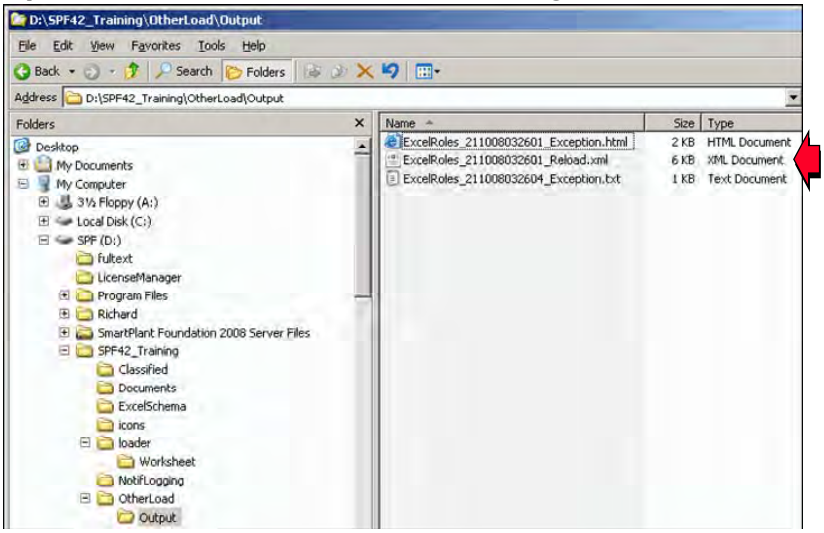
Reload files contain load file entries that have failed in NLF format so that they can be fixed and then reloaded. The reload file will contain failed containers

The name contains original load file's name, a date/time stamp, and the word Reload. For example:

> ExcelRoles.xml generates reload file named ExcelRoles_211008032601_ Reload.xml or similarly for an NLF file ExcelRoles.dwh generates reload file named ExcelRoles_ 200105095938_Reload.dwh

Loader Reload Files

An Output folder is created to contain the generated reload files.



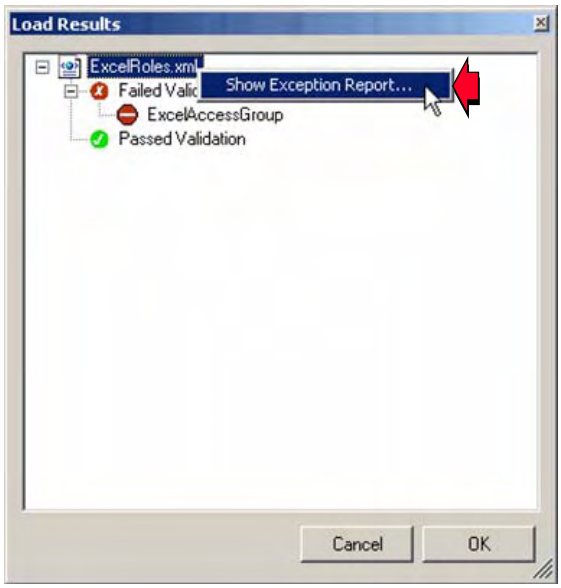
© 2005. Intergraph Corp.
All Rights Reserved.

An exception reports consists of an XML file that contains exceptions reporting information and an XSL file to control formatting of the exceptions report. Both of these files are placed in the Output directory.

If an exceptions report is created, the resulting file can be viewed by selecting the shortcut menu option Show Exception Report on the file node in the Load Results dialog box.

Loader Reload Files

- **Right-click on the XML file and select *Show Exception Report* from the right mouse menu.**

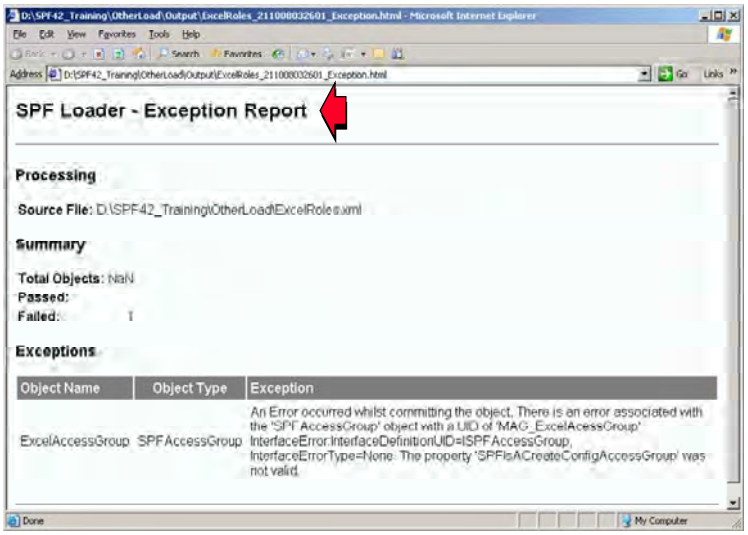


© 2005. Intergraph Corp.
All Rights Reserved.

The following is an example of an exceptions report generated when an error loading a file occurs.

Loader Reload Files

A browser window is displayed with the results of the report.



© 2005. Intergraph Corp.
All Rights Reserved.

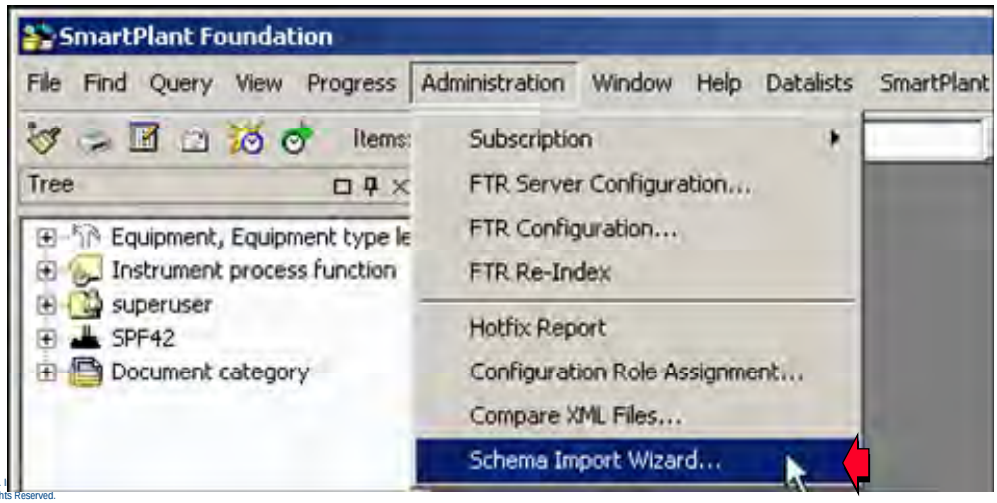
2.6 Introduction to the Schema Import Wizard

The Schema Import Wizard can be used to import schema and administration files into the SPF database but not data files. Launch the Schema Import Wizard from the Desktop Client to accomplish this task.

To import schema data:

Importing Schema Additions

- **In the SmartPlant Foundation Desktop Client, choose *Administration > Schema Import Wizard*.**

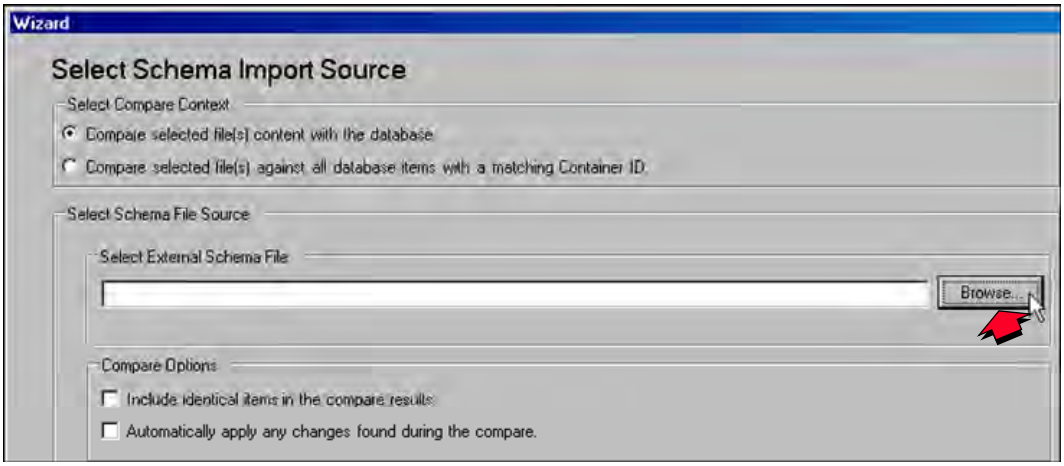


© 2005. I
All Rights Reserved.

The import *Wizard* form will appear.

Importing Schema Additions

- **In the *Schema Import Wizard*, click the *Browse* button to search for the schema file to be loaded.**

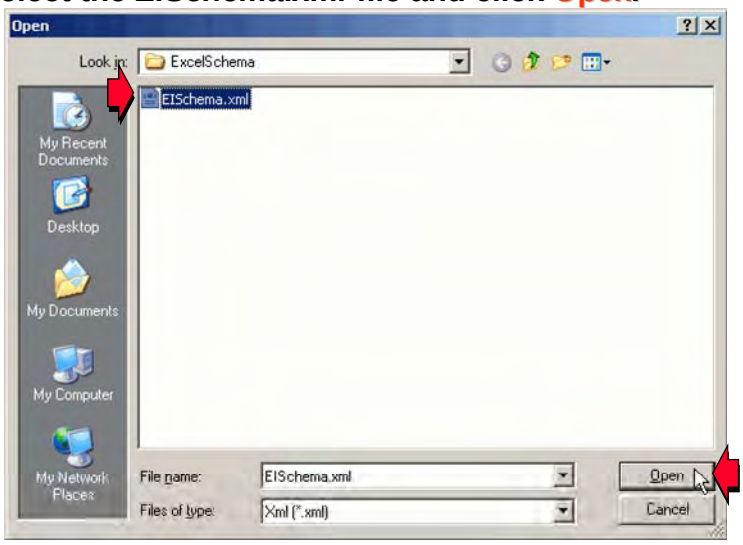


© 2005. Intergraph Corp.
All Rights Reserved.

When the browser dialog appears, select the schema file to be imported.

Importing Schema Additions

- **Select the *EISchema.xml* file and click Open.**

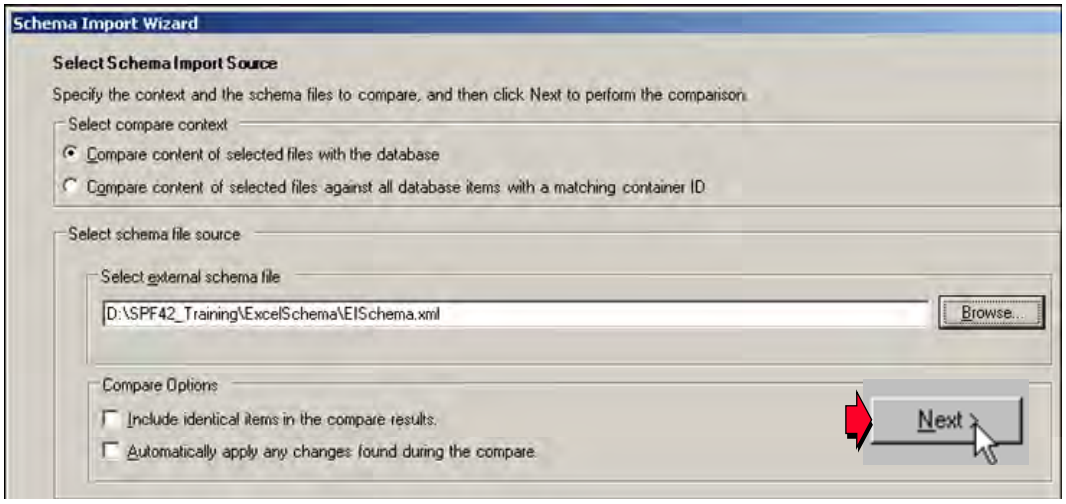


© 2005. Intergraph Corp.
All Rights Reserved.

When importing a schema into SmartPlant Foundation, it is necessary to first compare the schema with the existing content of the database and then selectively load the updates.

Importing Schema Additions

- **Click Next to compare schema file content with the database.**



Below are the selectable options on the *Schema Import Wizard* form:

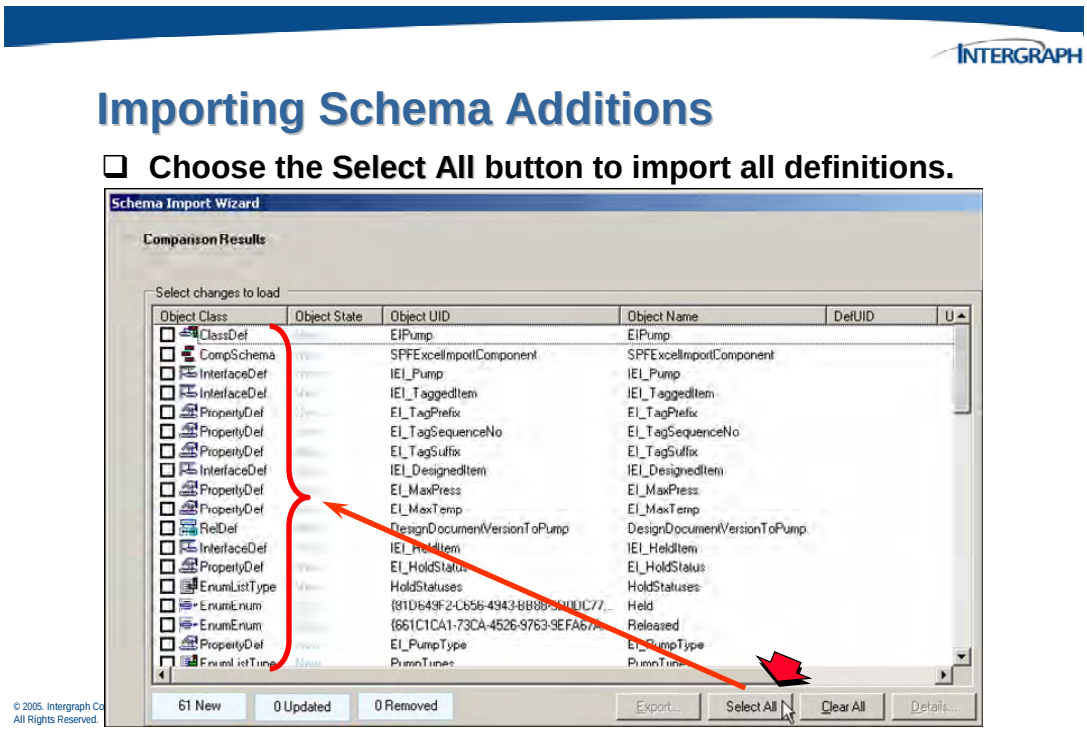
Compare selected file(s) content with the database – each object in the file is queried in the database and all new, modified or deleted objects and relationships in a list view in the *Comparison Results* form.

Compare selected file(s) against all database items with a matching ContainerID - the wizard reads the header of the selected schema file for the ContainerID. It then queries the SmartPlant Foundation database for all schema items with that ContainerID and presents all new, modified or deleted objects and relationships in a list view in the *Comparison Results* form.

Include identical items in the compare results – the compare results include identical objects.

Automatically apply any changes found during the compare – will automatically load any schema changes found into the SmartPlant Foundation database.

The *Comparison Results* form will appear. To selectively load updates, enable the corresponding checkboxes next to the object to be loaded.



The totals (*New*, *Updated* and *Removed* objects and relationships) are displayed at the bottom of the form.

The following describes the fields on the *Comparison Results* **Select changes to load** form:

- **Object Class** - Displays the classification of the object.
- **Object State** - Displays whether the object is new, updated, or removed.
- **Object UID** - Displays the object's unique identifier.
- **Object Name** - Displays the object name.
- **New** - Displays the number of new objects loaded.
- **Updated** - Displays the number of updated objects loaded.
- **Removed** - Displays the number of objects removed.
- **Identical** - Displays the number of objects found that are identical to the database source.
- **Export** - Exports the compared objects to a file for review.
- **Select All** - Selects all objects to load.

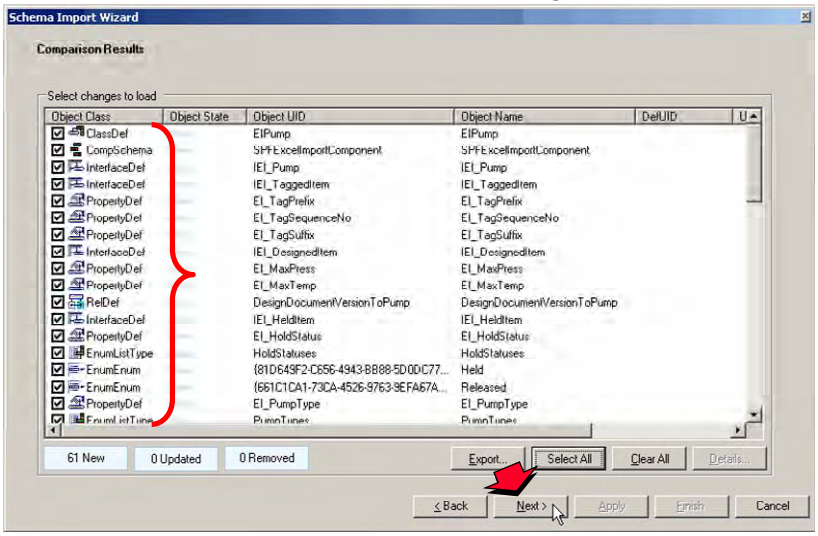
- **Clear All** - Removes all objects from the load.
- **Details** - Displays the load results in a text file.
  - **Added** - Displays any new objects or properties in blue.
  - **Changed** - Displays updated objects or properties in yellow.
  - **Removed** - Displays any deleted objects or properties in red.

If an object fails to load, place your mouse over the object to display details about the exception.

The Schema Import Wizard will reorder strings in the text file to match those in the database. These will be displayed as updated objects, but the function of the objects does not change.

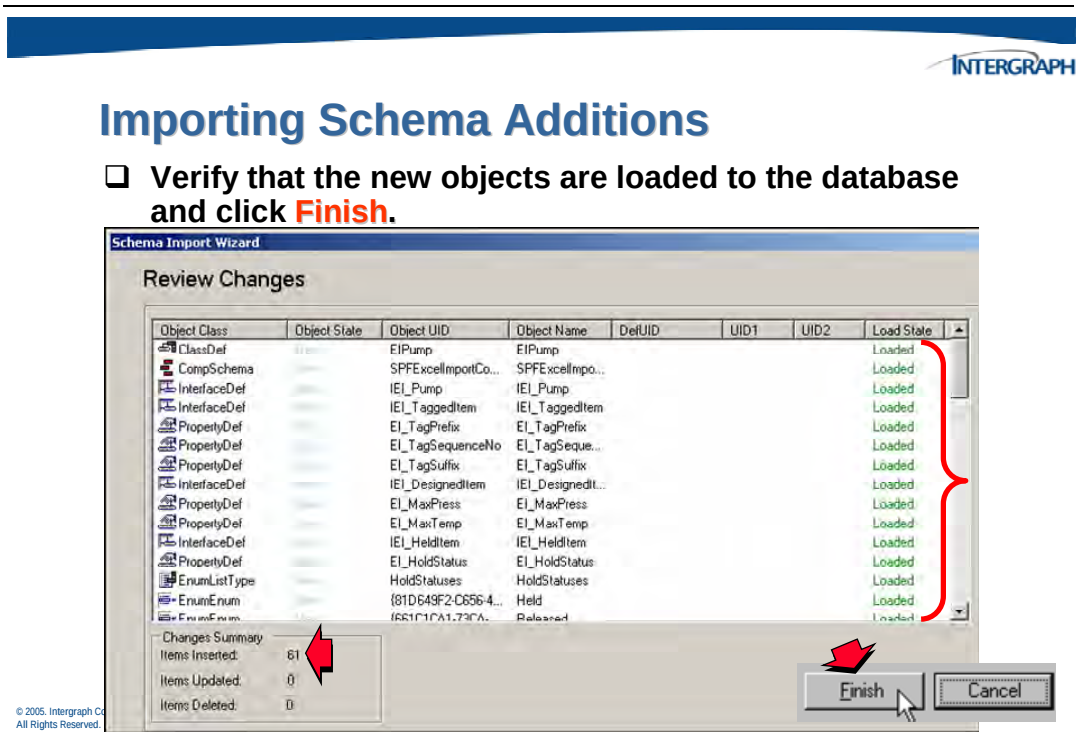
Importing Schema Additions

- **Click the Next button to start the import.**



© 2005. Intergraph Corp.
All Rights Reserved.

After the update, the wizard displays the load status for each object and a summary in the *Review Changes* form.



The following describes the fields on the *Review Changes* form:

- **Object Class** - Displays the classification of the object.
- **Object State** - Displays whether the object is new, updated, or removed.
- **Object UID** - Displays the object's unique identifier.
- **Object Name** - Displays the object name.
- **Load State** - Displays

Changes Summary

- **Items Inserted** - Displays the number of new items inserted into the database.
- **Items Updated** - Displays the number of items updated in the database.
- **Items Deleted** - Displays the number of items removed from the database.

2.7 Activity 1 – Loading SPF Users, Organizations and Role Assignments

Complete **Chapter 2 - Activity 1** in the SPF Configuration and Administration II activity workbook.

2.8 Activity 2 – Create Design Document Load File

Complete **Chapter 2 - Activity 2** in the SPF Configuration and Administration II activity workbook.

2.9 Activity 3 – Loading Schema Definitions with the Schema Import Wizard

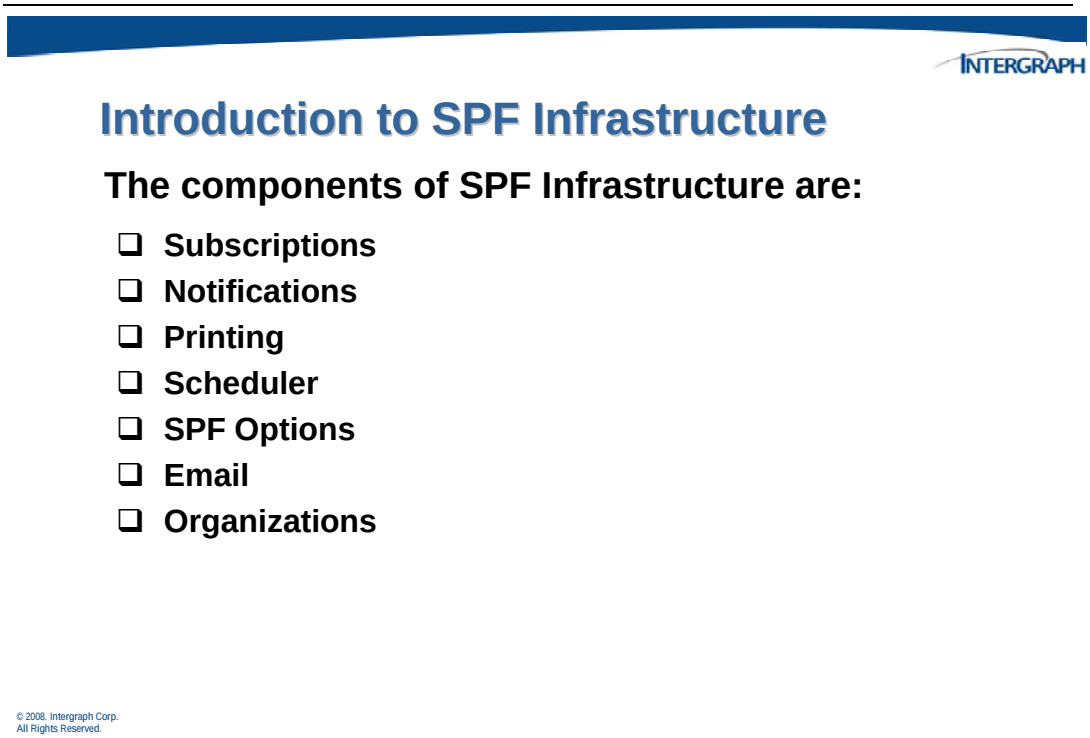
Complete **Chapter 2 - Activity 3** in the SPF Configuration and Administration II activity workbook.

CHAPTER 3

SPF Infrastructure Objects

3. Introduction to SPF Infrastructure

SmartPlant Foundation infrastructure includes support for such tasks as subscriptions and notifications, printing, the scheduler task, SPF options, email and organizations (companies).



To set up and configure the infrastructure objects, the *Desktop Client* is used.

3.1 Container Management

The **container** of an object or relationship is just a convenient label used to group together sets of objects and relationships. Each of the source SmartPlant Foundation build files have a different container so that all objects and relationships loaded from a single file have the same container. All the schema and administration objects and relationships delivered with the system have a **container ID** that users can view on the object details and relationship forms.

INTERGRAPH

Containers

Containers segment components of the SPF configuration.

SPF is built from sets of files.

Each object and relationship *can* have a containerID, and is used to separate the data into sub-groups.

Usually, the following objects have containerIDs

- **ADMIN and SCHEMA (domains)**
- **Reference Data (for example, classification trees)**

Note: ContainerIDs can cross domains, thus Authoring appears in both the ADMIN and SCHEMA domains.

© 2008. Intergraph Corp.
All Rights Reserved.

The container can be any string, but typically is made up of two strings concatenated with a full stop. The first part represents a larger grouping or layer of the configuration and the second part represents a specific bit of functionality within that layer. For example:

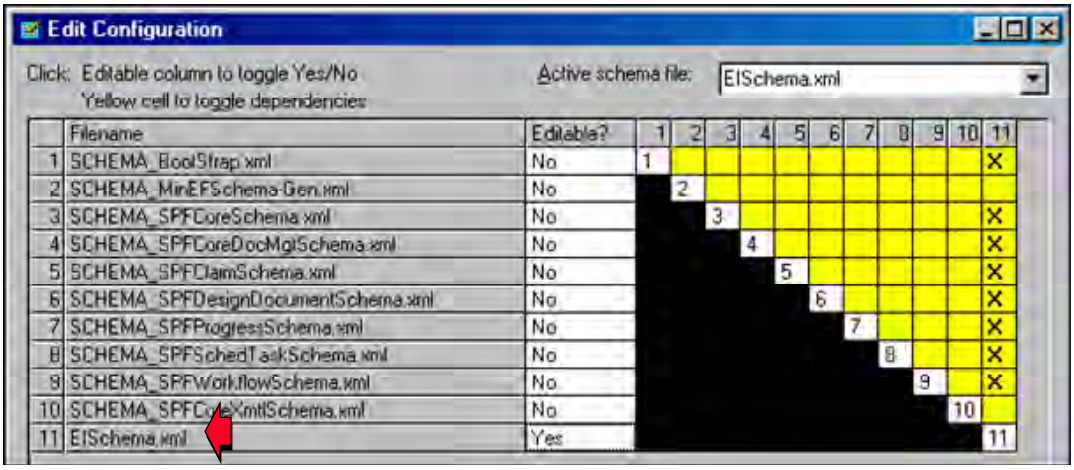
- Core.DocControl
- Core.DocMgt
- PlantAndProject.WBS

When creating new objects and relationships, the container is set to the one specified in the user preferences.

In the model used for this training, a custom class, interfaces, properties and relationships have been added to the delivered model. These new additions are managed in their own container. To add the new extensions, use a delivered configuration file, \SmartPlant\Foundation\2008\Models\Core\900-AllCore.cfg, along with a custom extension file, **EISchema.xml**. Use the *Schema Editor* to make the additions to the model.

Custom Schema Extensions

Extend the delivered system with a schema extensions file containing a new class, interfaces, properties, enumerated lists and relationships.

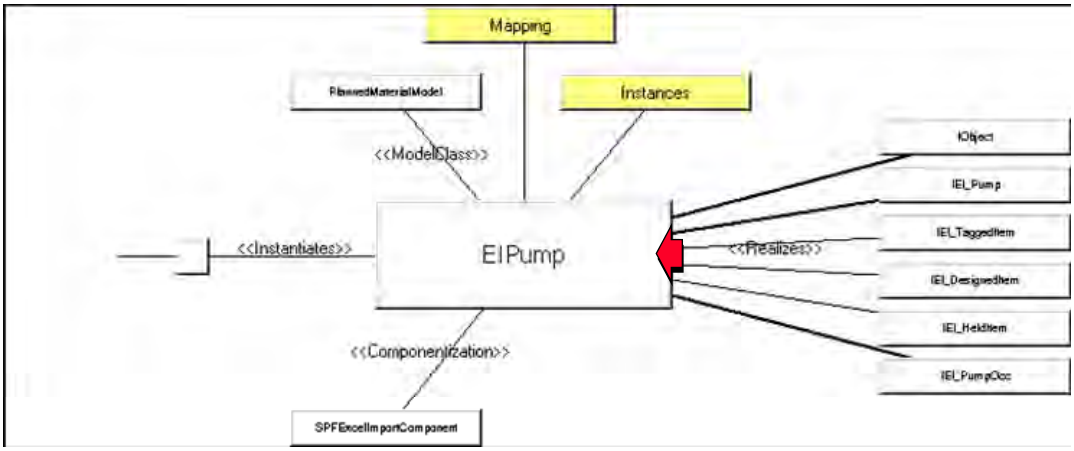


	Filename	Editable?	1	2	3	4	5	6	7	8	9	10	11
1	SCHEMA_BoolStrap.xml	No	1										X
2	SCHEMA_MinEFSchema-Gen.xml	No		2									
3	SCHEMA_SPFCoreSchema.xml	No			3								X
4	SCHEMA_SPFCoreDocMgtSchema.xml	No				4							X
5	SCHEMA_SPFClaimSchema.xml	No					5						X
6	SCHEMA_SPFDesignDocumentSchema.xml	No						6					X
7	SCHEMA_SPFProgressSchema.xml	No							7				X
8	SCHEMA_SPFSchedTaskSchema.xml	No								8			X
9	SCHEMA_SPFWorkflowSchema.xml	No									9		X
10	SCHEMA_SPFCoreXmtlSchema.xml	No										10	
11	EISchema.xml	Yes											11

This *Tree/UML* view diagram will show the new *ClassDef* that has been modeled.

Custom Class and Interfaces

A new class called *EIPump* will has been modeled and loaded into the SPF database.

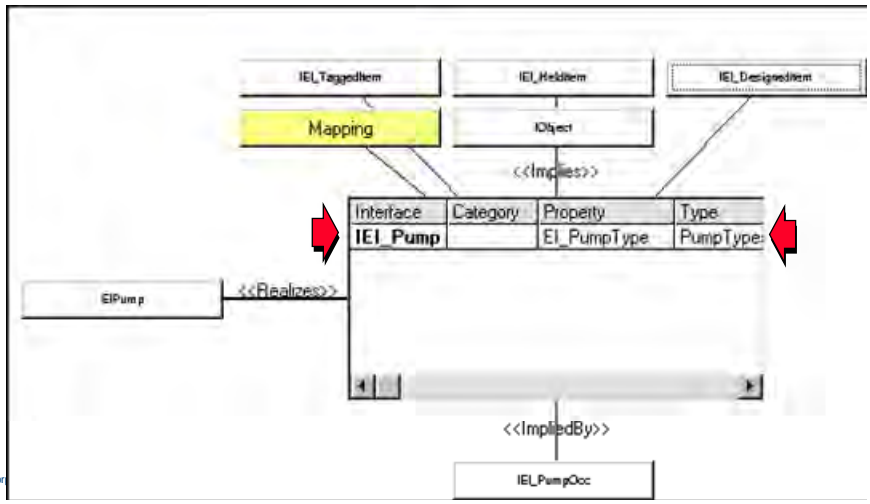


© 2008. Intergraph Corp.
All Rights Reserved.

This *Tree/UML* view diagram will show one of the new *InterfaceDef* that has been modeled. The property exposed by this interface is scoped by an enumerated list.

Custom Class and Interfaces

The primary interface for the new class is IEI_Pump which exposes the property *EI_PumpType*.

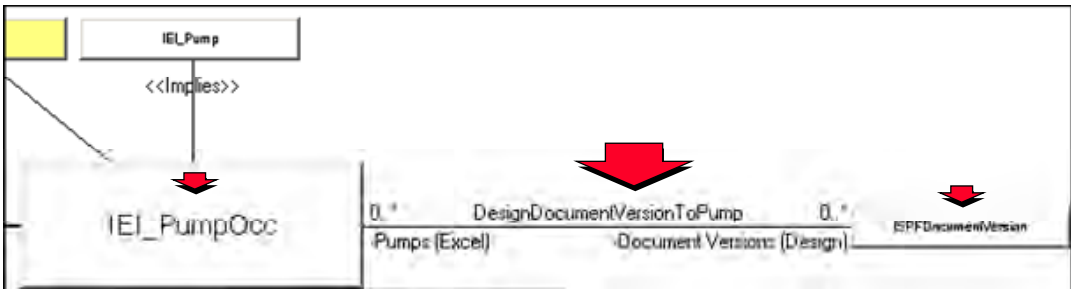


© 2008. Intergraph Corp.
All Rights Reserved.

This *Tree/UML* view diagram will show a new *RelDef* that has been modeled.

Custom Class and Interfaces

The interface IEI_Pump is related to ISPFDocumentVersion with the relationship definition *DesignDocumentVersionToPump*.

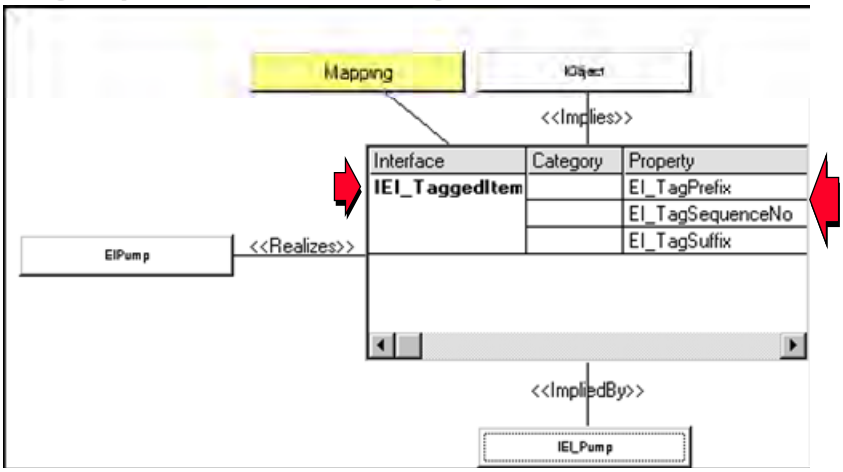


© 2008. Intergraph Corp.
All Rights Reserved.

This *Tree/UML* view diagram will show another of the new *InterfaceDefs* that has been modeled.

Custom Class and Interfaces

Another realized interface for the class is IEI_TaggedItem which exposes the properties *EI_TagPrefix*, *EI_TagSequenceNo*, and *EI_TagSuffix*.

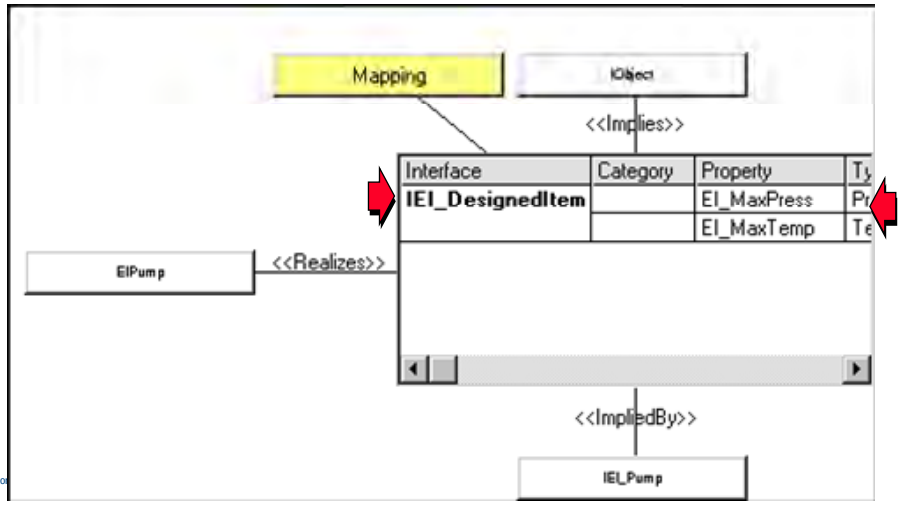


© 2008. Intergraph Corp.
All Rights Reserved.

The properties exposed by this interface are scoped by UoM's.

Custom Class and Interfaces

Another realized interface for the class is IEI_DesignedItem which exposes the properties *EI_MaxPress*, and *EI_MaxTemp*.

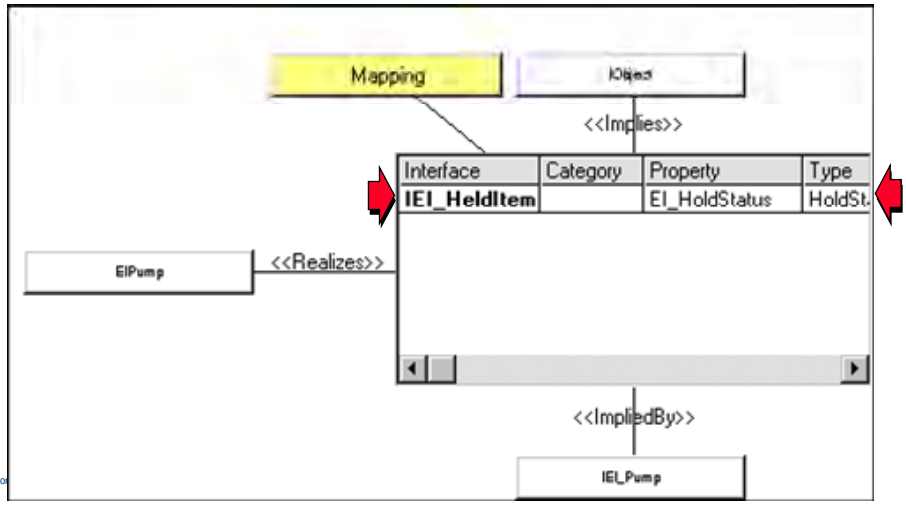


© 2008. Intergraph Co
All Rights Reserved.

The property exposed by this interface is also scoped by an enumerated list.

Custom Class and Interfaces

Another realized interface for the class is IEI_HeldItem which exposes the property *EI_HoldStatus*.

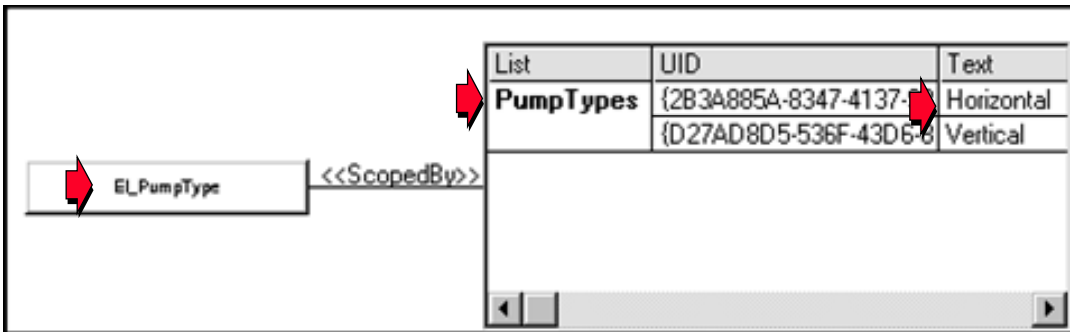


© 2008. Intergraph Co
All Rights Reserved.

This *Tree/UML* view diagram shows the values contained in the enumerated list.

Custom Enumerated Lists

The property *EI_PumpType* is scoped by the enumerated list PumpTypes.

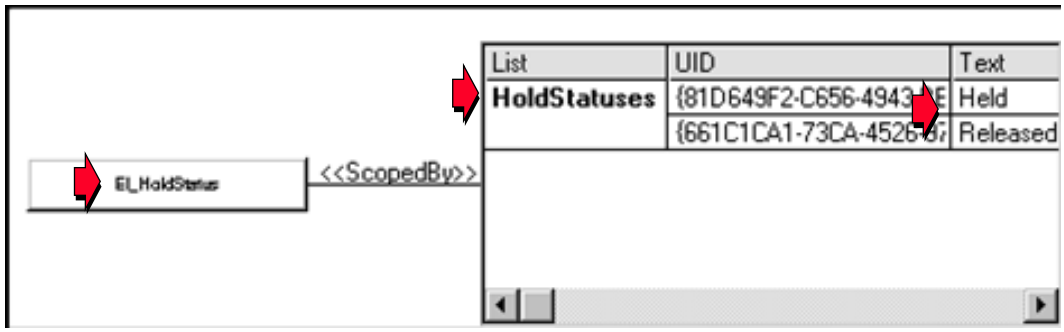


© 2008. Intergraph Corp.
All Rights Reserved.

This *Tree/UML* view diagram shows the values contained in the enumerated list.

Custom Enumerated Lists

The property *EI_HoldStatus* is scoped by the enumerated list HoldStatuses.



© 2008. Intergraph Corp.
All Rights Reserved.

The schema extensions shown above are grouped in their own container, **Trn.Mods**.

Containers

Containers segment components of the SPF configuration.

SPF is built from sets of files.

Each object and relationship *can* have a containerID, and is used to separate the data into sub-groups.

Usually, the following objects have containerIDs

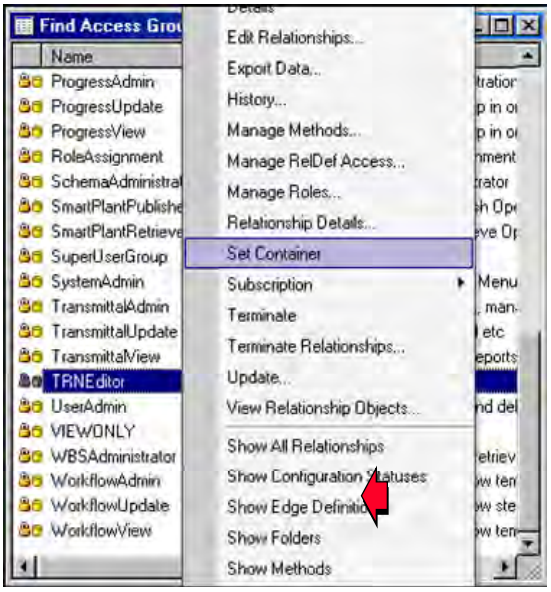
- **ADMIN and SCHEMA (domains)**
- **Reference Data (for example, classification trees)**

© 2008. Intergraph Corp.
All Rights Reserved.

A **Container ID** can also be set for existing data. This can be done by a shortcut menu command on *Admin*, *Schema* or *Relationship* objects.

Container Management

- **Right click on *TRNEditor* and from the shortcut menu, select *Set Container*.**

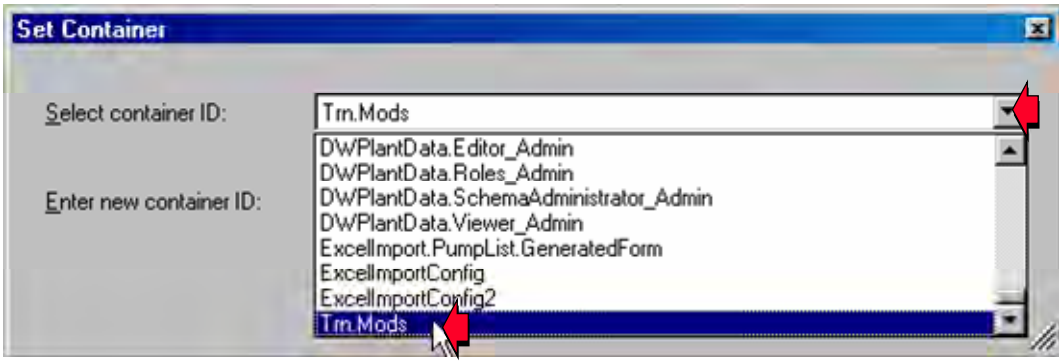


© 2008. Intergraph Corp.
All Rights Reserved.

Either select an existing ContainerID or type a new one.

Container Management

- **In the *Select container ID field*, choose a container name from the list of existing containers.**

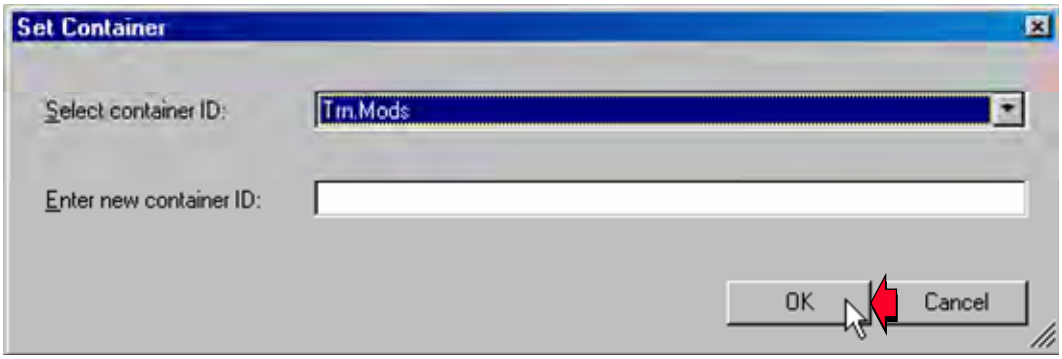


© 2008. Intergraph Corp.
All Rights Reserved.

Once a ContainerID has been set, the user preferences will save this information.

Container Management

- **Click OK to relate the selected Access Group object to this container.**

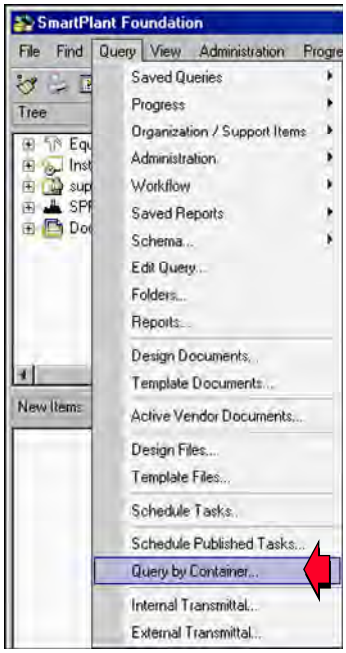


© 2008. Intergraph Corp.
All Rights Reserved.

Users can also perform queries by Container ID.

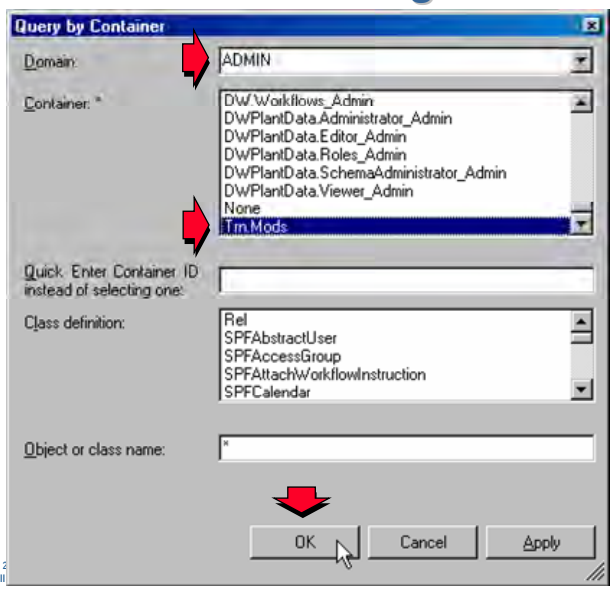
Container Management

- **Select *Query > Query by Container* from the menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

Container Management

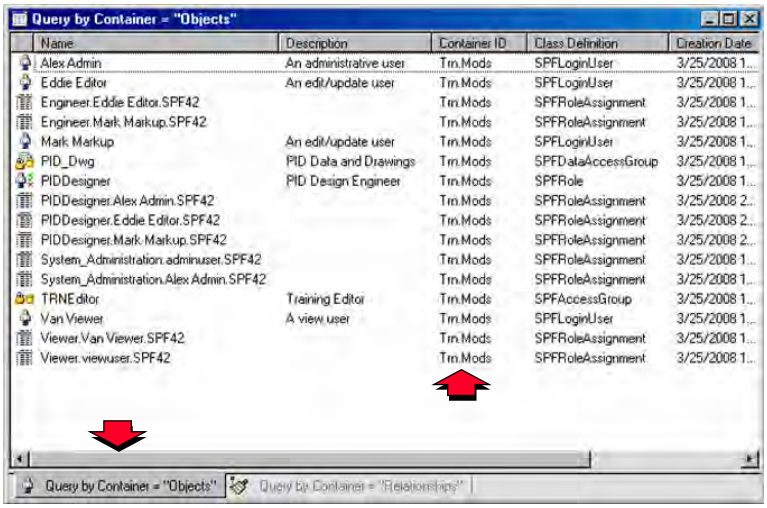


- **Choose an optional *Domain*, a required *Container* name and click OK to perform the query.**

You can filter by *domain*, selected *containers*, container wildcards, and the *class of object* to retrieve. To retrieve objects with no container, select a container of **None**.

Container Management

The Object contents of the container will display in one tab in the list view.



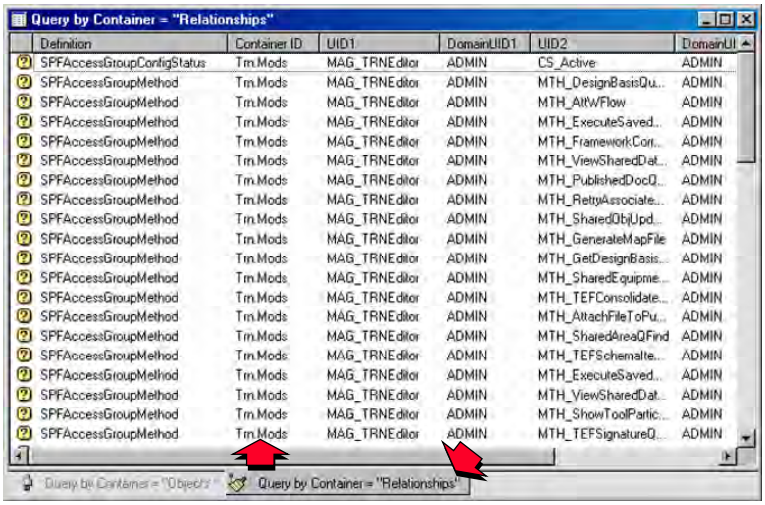
© 2008. Intergraph Corp.
All Rights Reserved.

Use multi-select, and then click ***Set Container*** on the shortcut menu on any object or in the list view.

For relationships, use the ***Set Container*** shortcut menu command on any of the relationship objects.

Container Management

- **Select the Relationships tab to view the relationship contents of the container.**

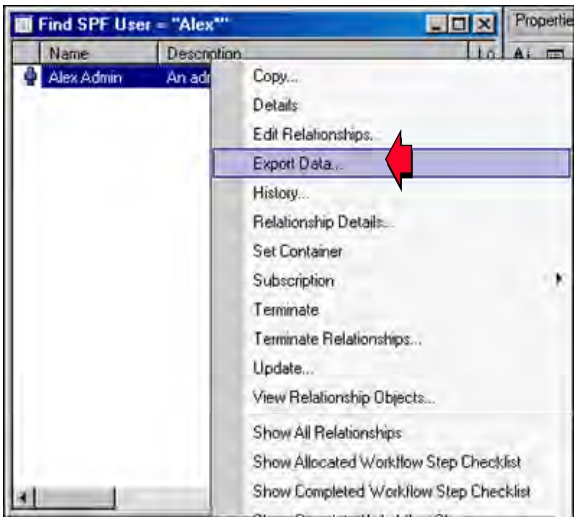


© 2008. Intergraph Corp.
All Rights Reserved.

Object definitions can be exported and segregated by Container ID.

Container Management

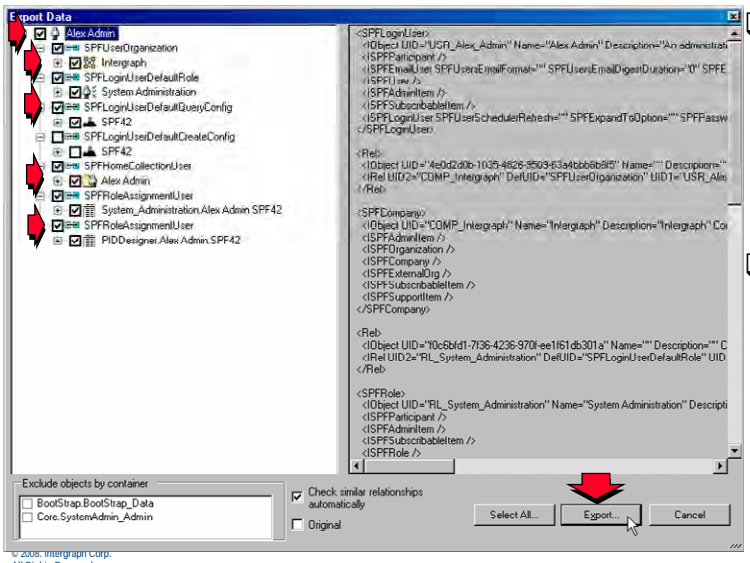
- **Right click on an object and from the shortcut menu, select *Export Data*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Export Data* form is displayed.

Container Management



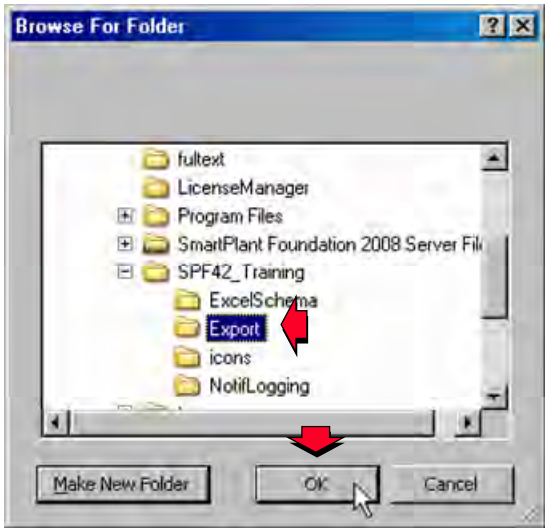
- Enable the appropriate check boxes in order to export these objects.
- Click the **Export** button to execute the operation.

© 2008. Intergraph Corp.
All Rights Reserved.

A *Browse For Folder* dialog appears.

Container Management

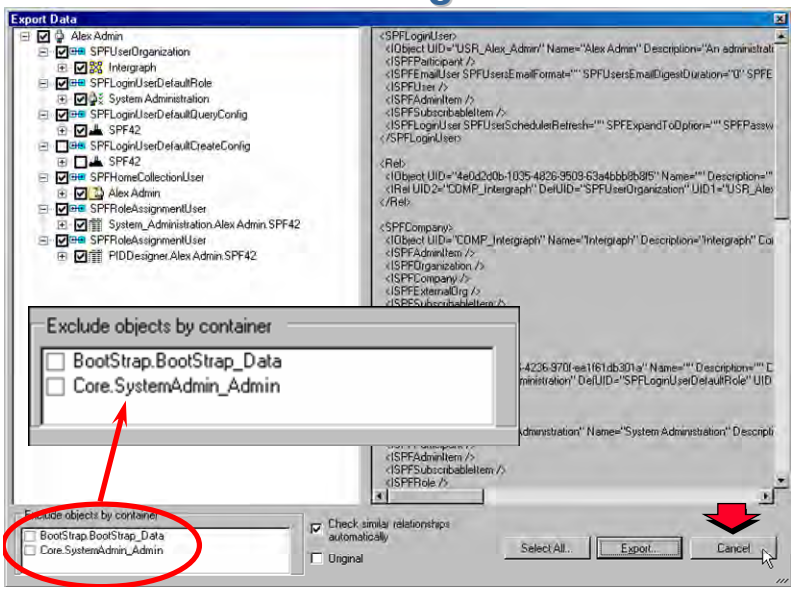
- Browse to choose a destination folder for the export results.



© 2008. Intergraph Corp.
All Rights Reserved.

You can use the *Exclude objects by container* box to filter the data to be exported.

Container Management

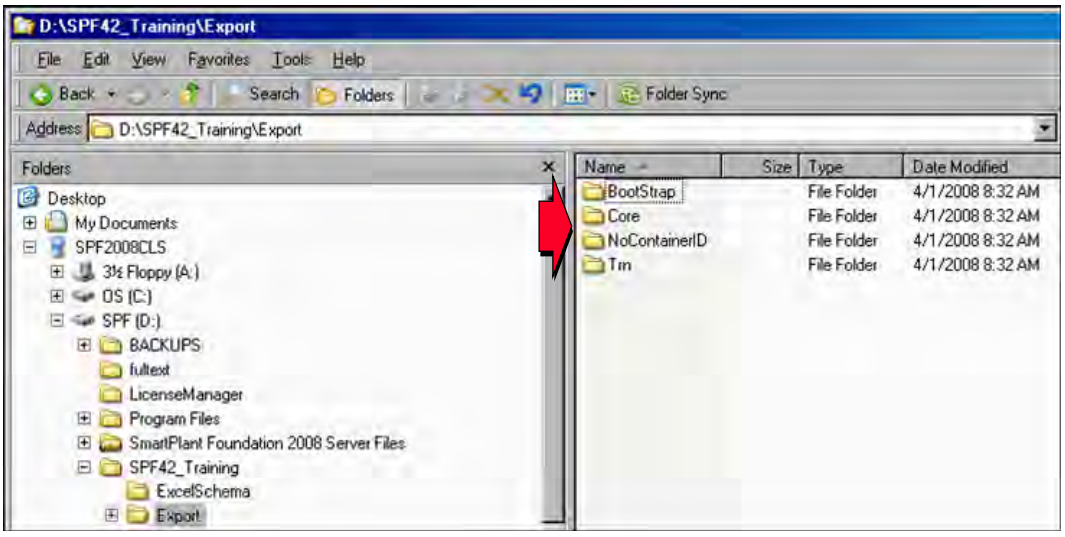


- **Choose the Cancel button to dismiss the *Export Data* form.**

When you export by container, the system creates a file for each container within a folder for the layer.

Container Management

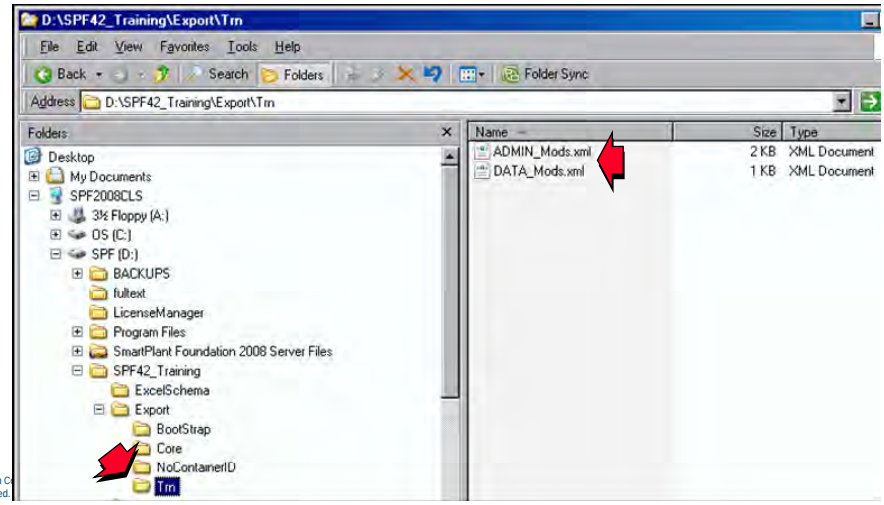
- **Use Windows Explorer to view the results of the export.**



The system also creates a file for *ADMIN_*, *SCHEMA_* and *DATA_* within each container folder.

Container Management

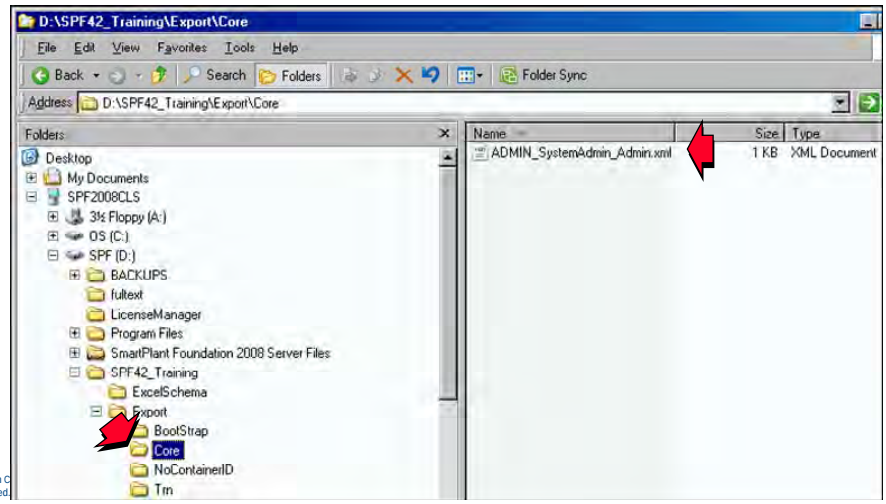
- **Double click on one of the *container* folders to view the contents.**



© 2008. Intergraph C
All Rights Reserved.

Container Management

- **Double click on another *container* folder to view the contents.**



© 2008. Intergraph C
All Rights Reserved.

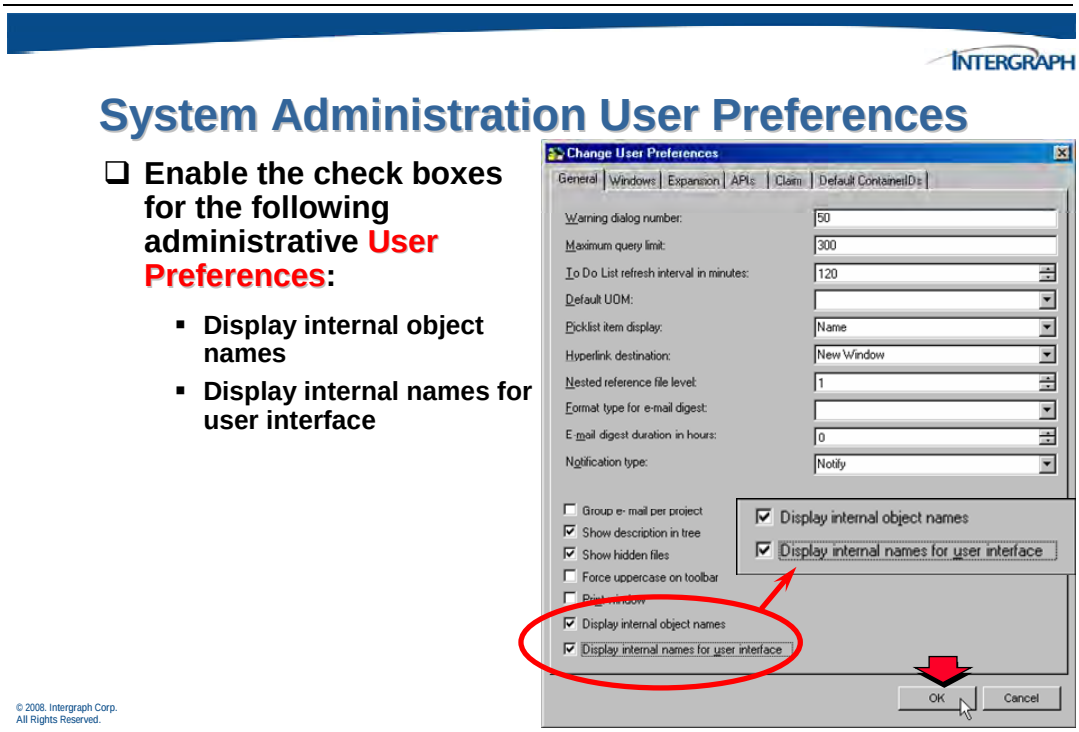
3.2 System Administration User Preferences

Various administrative user preferences are available to assist in the management of the GUI.

When configuring the menus and forms user interface, it is often difficult to track down the exact menu item that results in a menu or shortcut menu. There is an option to switch the GUI to display menu items using internal names.

This option is presented on the *User Preferences* dialog box and when selected, it does the following:

- Changes the menus items and form display items to be displayed using their internal name as opposed to *ISPFGUIDisplay.SPFGUIDisplayAs*



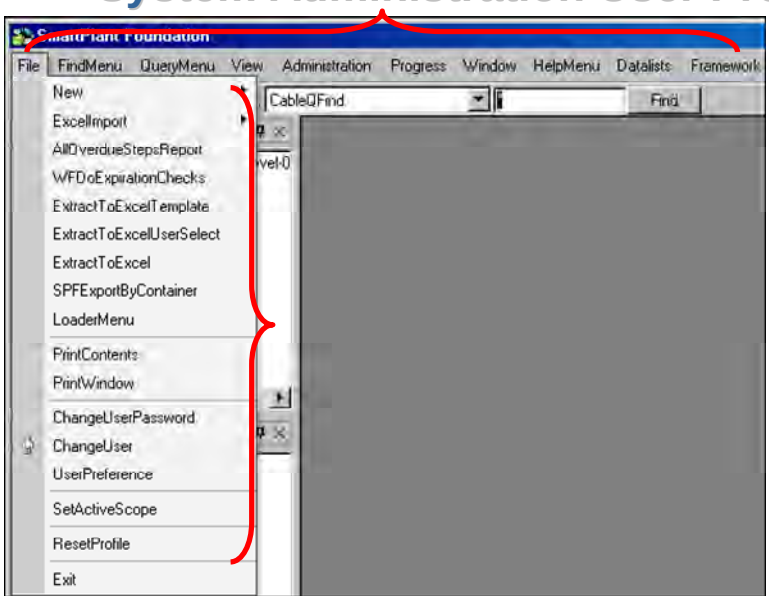
You must log out and back in to see the changes to the menus as these are cached. The shortcut menus and forms are affected immediately.

3.2.1 Display Objects Using Internal Names

By default, all data objects are presented to the user using the display name text taken from the object definition or the internationalization resource file. On some occasions, it is necessary to identify the objects by their internal name, and there is an option for this.

In the following example, the internal names are shown on the menus.

System Administration User Preferences

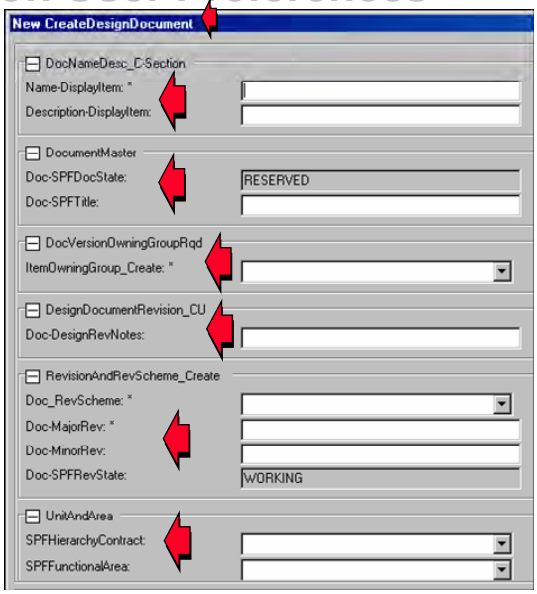


- **Select the *File* menu command to see the internal names displayed.**

Below are examples of the internal names are shown on forms.

System Administration User Preferences

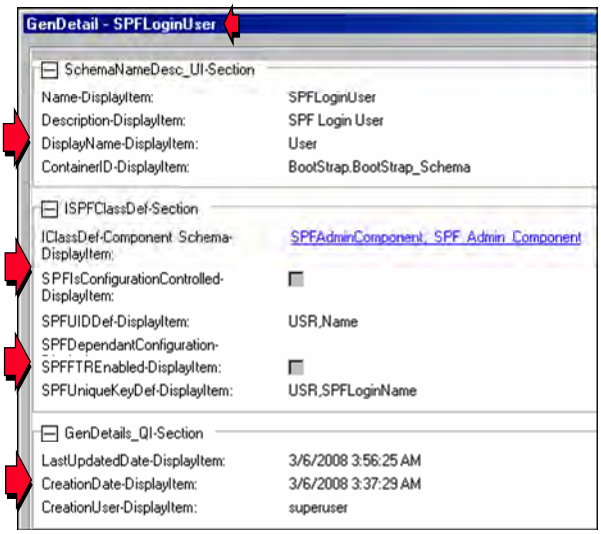
Note the internal names displayed in the form.



© 2008. Intergraph Corp.
All Rights Reserved.

System Administration User Preferences

Note the internal schema names displayed in this form.



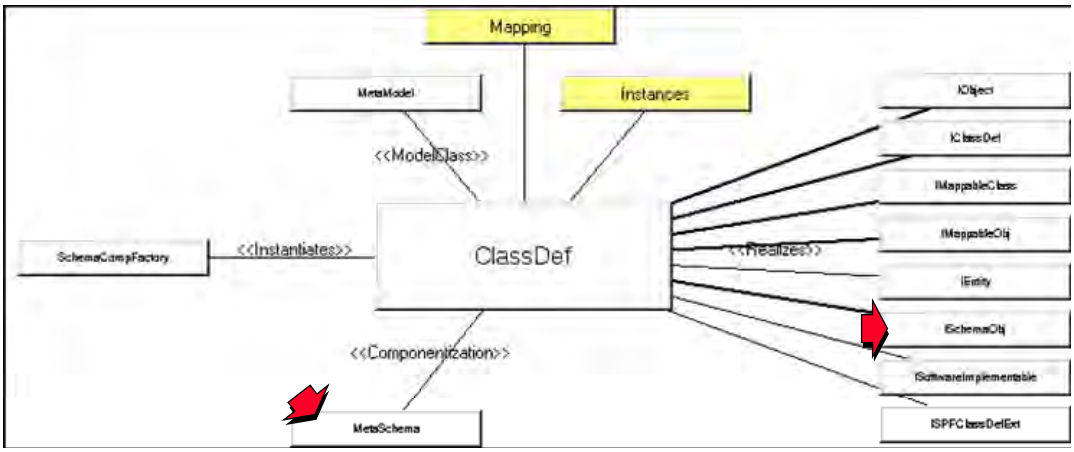
© 2008. Intergraph Corp.
All Rights Reserved.

The system displays **Schema** objects in list views and pick lists as follows:

- Display the *Name* by default.
- Display *ISchemaObj.DisplayName* defined in preferences if present.
- The *Display internal object names* (from preferences) overrides the ISchemaObj.DisplayName and any internationalization and displays the original Name on IObject.

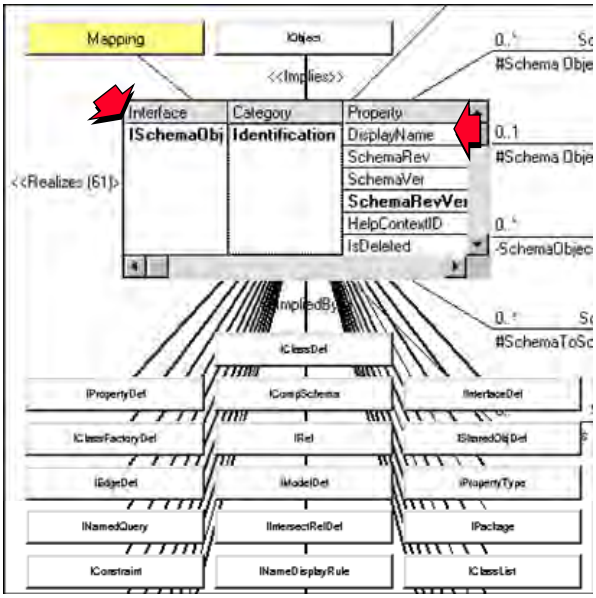
System Administration User Preferences

The system displays schema objects using the MetaSchema *ISchemaObj.DisplayName* interface/property defined in preferences if present.



System Administration User Preferences

DisplayName **is a property exposed by ISchemaObj.**



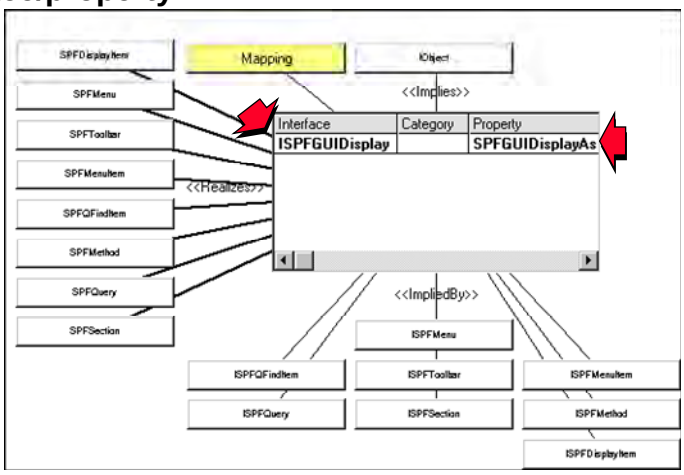
© 2008. Intergraph Corp.
All Rights Reserved.

The system displays **Data** and **Administration** objects in list views and pick lists as follows:

- Display the *Name* by default.
- Display *ISPFDisplayOverride.NameOverride* defined in preferences if present.
- ISPFGUIDisplay.SPFGUIDisplayAs is used to define the text to be displayed in the GUI, such as the menu item text or the display item text.
- The *Display internal object names* (from preferences) overrides the ISPFDisplayOverride.NameOverride and any internationalization and displays the original Name on IObject.

System Administration User Preferences

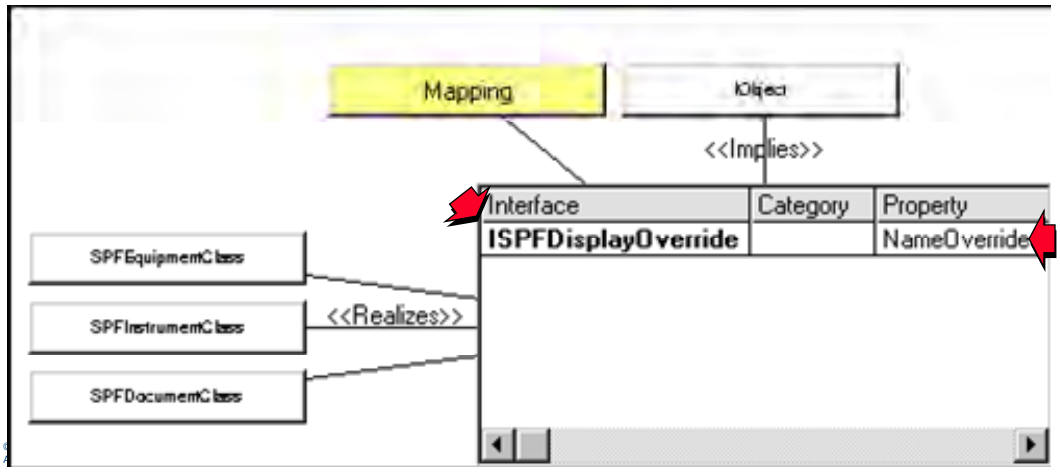
The system displays menu item text or display item text using the MetaSchema *ISPFGUIDisplay.SPFGUIDisplayAs* interface/property.



© 2008. Intergraph Corp.
All Rights Reserved.

System Administration User Preferences

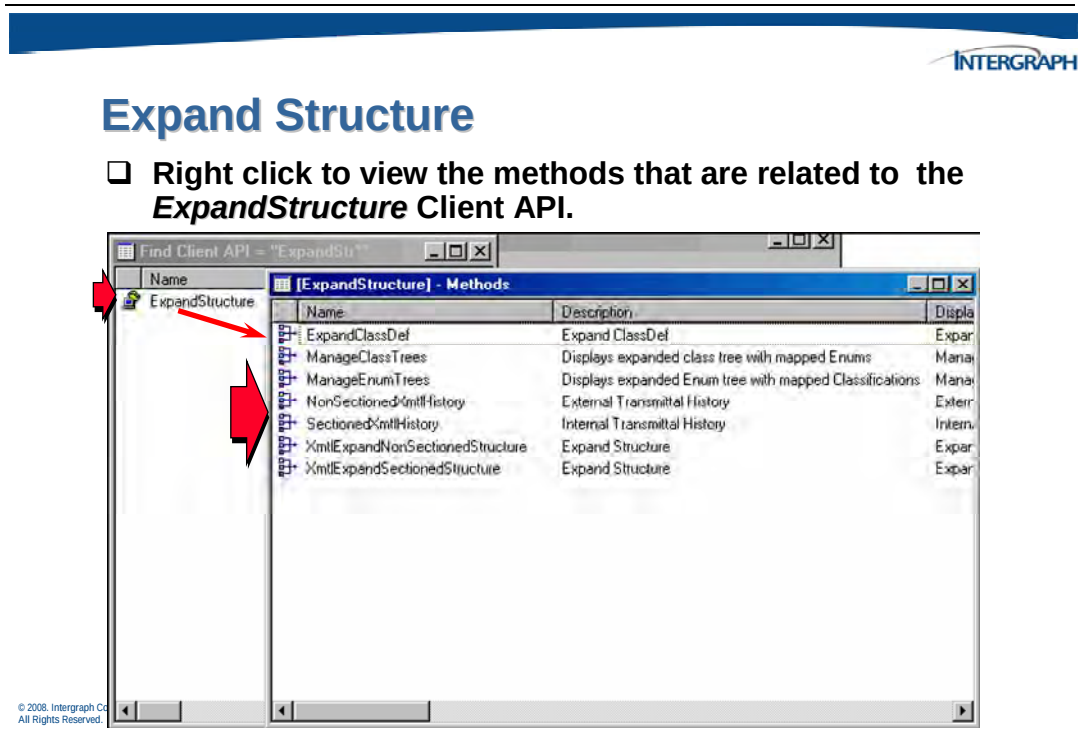
The system displays data or administration objects using the MetaSchema *ISPFDisplayOverride.NameOverride* interface/property defined in preferences if present.



3.3 Expand Structure

The client API ExpandStructure allows the user to display the structure of a selected object. It can be found on trees, class definitions, and transmittals.

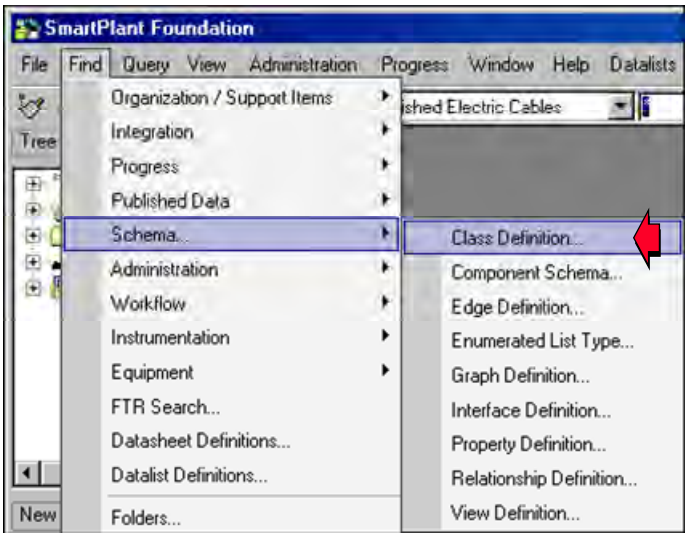
The **ExpandStructure** client API uses the following methods to expand:



The following is an example of the ExpandStructure for a *ClassDef*.

Expand Structure

- **From the menu, select *Find > Schema > Class Definition*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Class Definition* dialog will appear.

Expand Structure

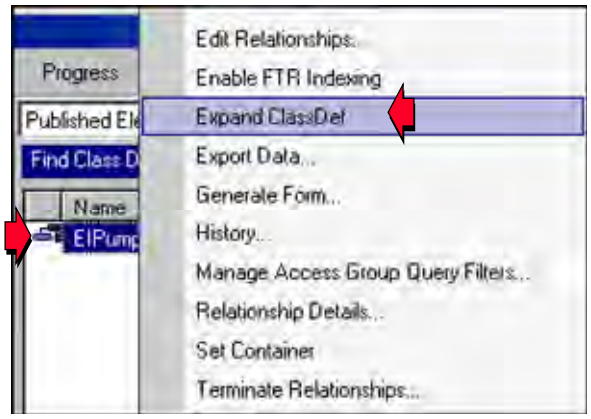
- **Enter *EIPump* * to find and display the pump class and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Select the ClassDef to expand from the list view window.

Expand Structure

- **Right click on the *EIPump* class and from the shortcut menu, select *Expand ClassDef*.**

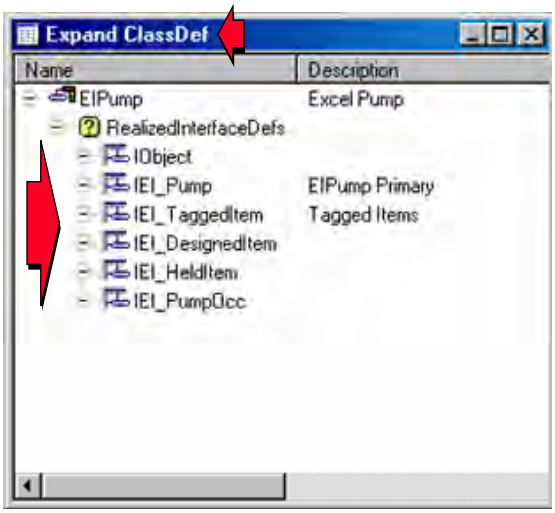


© 2008. Intergraph Corp.
All Rights Reserved.

The expanded ClassDef structure list view is displayed.

Expand Structure

The EIPump class and it's Realized interfaces will be displayed.



© 2008. Intergraph Corp.
All Rights Reserved.

3.4 Subscriptions and Notifications

In the Desktop Client, users can register their interest in specific events for objects or object classes to receive notifications when that event is performed for that object or class. Object subscriptions are where a user requests a notification when one or more events occur on an object under specified conditions.

INTERGRAPH

SPF Subscriptions and Notifications

Subscription is the process by which a specified task is executed once a subscribed event has occured.

This task can be:

- **Notification, either via *Email* or the *SPF Client Inbox***
- **Logging to a file**
- **External application DLL (stubbed in for future functionality)**

There are two basic types of subscription:

- **Object subscription**
- **Class subscription**

If a user subscribes to an object, they can register for email or SPF inbox notification when a particular command takes place on the object.

© 2008, Intergraph Corp.
All Rights Reserved.

Subscriptions can be created and managed at the *object* or *class* level. However, **log** subscriptions can only be created at class level.

Examples of events and subscriptions:

- To be notified when a selected document revision is signed off.
- To be notified when a selected document revision is signed off with a status of “Final”.

Subscriptions are configuration specific such that when the events occur on objects, the notifications are only sent in the configuration in which they were created.

SPF Subscriptions and Notifications

In object subscriptions the object has a relationship between the object and the subscription.

With class subscription you store the interface on the subscription. Otherwise the process is the same as for object subscriptions.

© 2008. Intergraph Corp.
All Rights Reserved.

Object subscriptions can be created for:

- The current user (Manage My Subscriptions).
- Other users or roles (Manage All Subscriptions to Object).

SPF Subscriptions and Notifications

A subscription definition object links one of the subscription mechanisms to the object or class of interest.

There are three predefined subscription classes to handle:

- **Notification**
- **Logging**
- **Running external DLLs (stubbed in for future functionality)**

All use the common ISubscriptionDef interface.

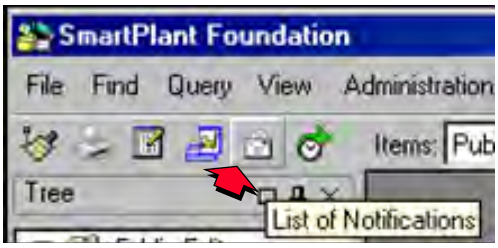
© 2008. Intergraph Corp.
All Rights Reserved.

3.4.1 Enabling Notification List Access

One way of viewing events that have been subscribed to is to use the *List of Notifications* toolbar command.

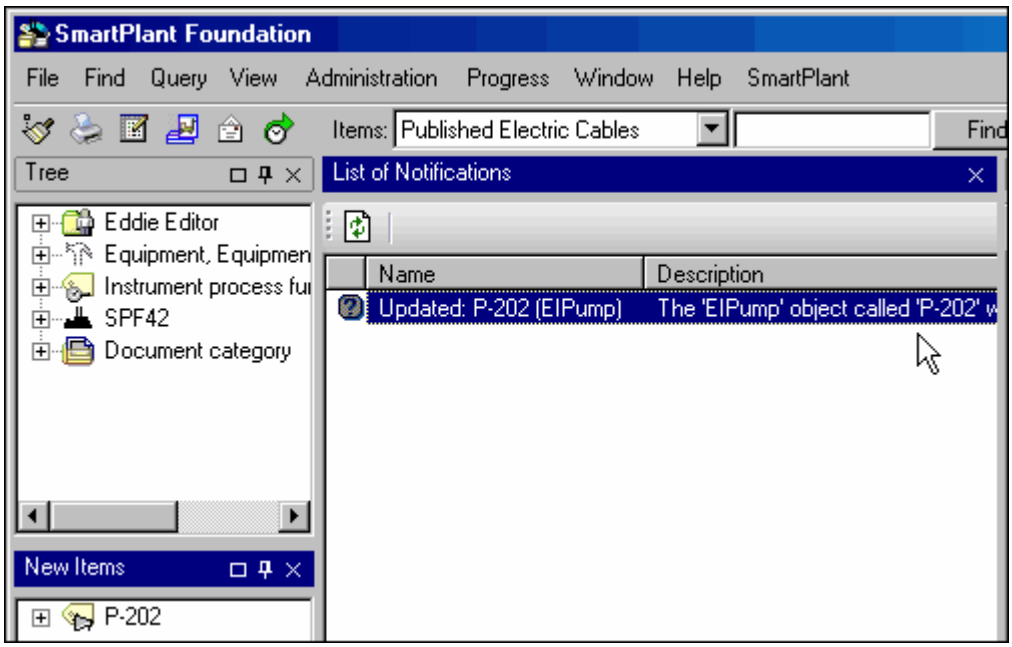
Setting Up Notification List Access

In order to view notifications, you must have access to the *List of Notifications* toolbar command.



© 2008. Intergraph Corp.
All Rights Reserved.

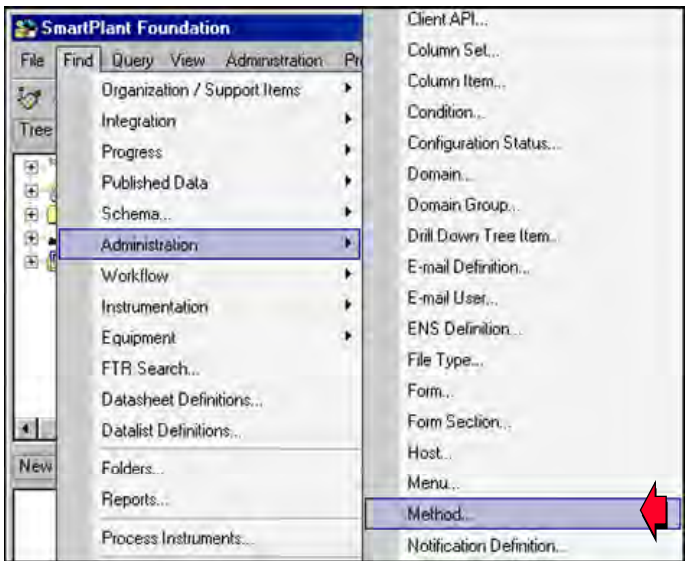
This command can be used to see the notifications that are generated in a list view window. This could be an alternative for using email.



If this toolbar command is not present in the user's Desktop Client, it can be easily configured by an administrator. Perform a search in order to locate the correct method, *NotificationList*. (Methods will be covered in detail later in this course in Chapter 5.)

Setting Up Notification List Access

- **Select *Find* > *Administration* > *Method* from the menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Method* dialog will appear.

Setting Up Notification List Access

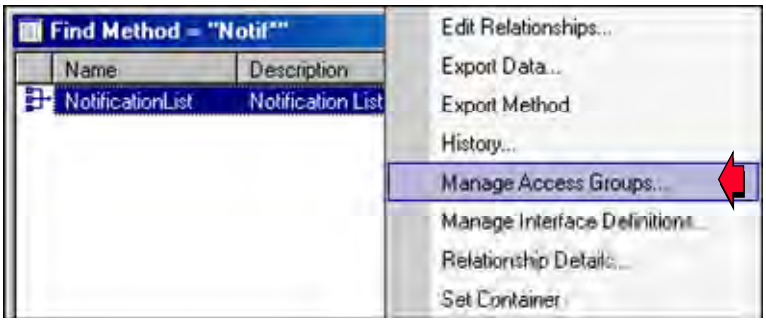
- **Enter *Notif* with the wildcard character (*) to find and display all methods that start with *Notif* and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Make sure that the *NotificationList* method is related to an **Access Group** that is available to a role assignment for the necessary users.

Setting Up Notification List Access

- **Right click on *NotificationList* and from the shortcut menu, select *Manage Access Groups*.**

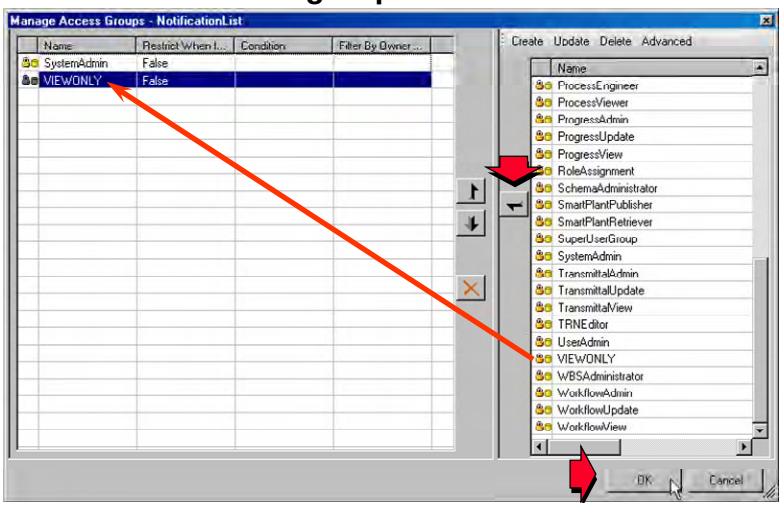


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Related Items* form will appear, in this case managing related access groups to the NotificationList method. Select the VIEWONLY group and then the left arrow to add a relationship to this access group.

Setting Up Notification List Access

- **Select the *VIEWONLY* Access Group and click the left arrow button to relate this group to the method and click OK.**



© 2008. Intergraph Corp.
All Rights Reserved.

By default in the initial release of SPF 2008, only the **SystemAdmin** access group is related to the *Notification Lis*t method. That means that only users with role assignments of SystemAdmin would have access to the method. By adding the **VIEWONLY** access group, many more users will now see this toolbar command in their Desktop Client window.

3.4.2 Setting Up Subscriptions

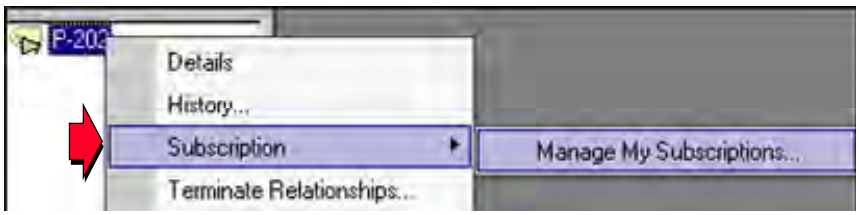
Although change notification is important for document management, you can subscribe to changes for other objects throughout the Desktop Client, provided that those objects are subscribable. Object subscriptions can be created from a RHM command, but ONLY on objects that realize the **ISPFSubscribableItem** interface.

INTERGRAPH

Setting Up Subscriptions

The ISPFSubscribableItem Interface

This interface must be realized by any class that you want to be subscribable. It is the interface that all the subscription RHM methods are attached to.



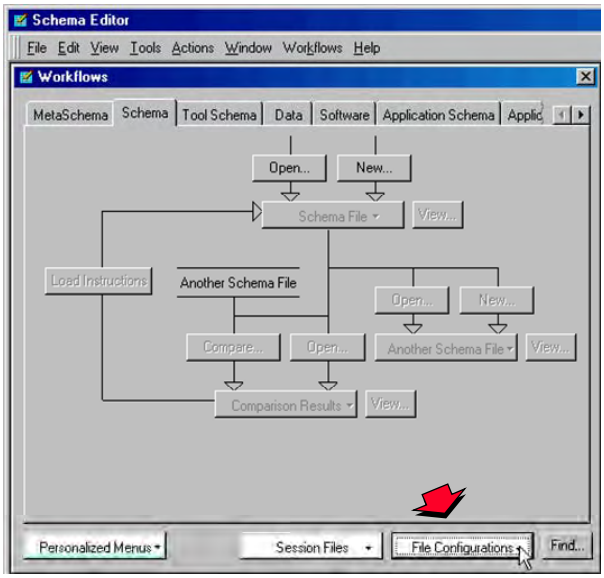
© 2008. Intergraph Corp.
All Rights Reserved.

In the following example, a custom class for a pump, **EIPump**, has been created for the plant configuration. By default, this custom class may not realize the **ISPFSubscribableItem** interface. In order to use subscriptions with this new class, minor schema modifications will need to be made in order to add the subscription functionality.

The first step is to use the schema editor to open the core schemas as well as the extension schema containing the custom class definition.

Setting Up Subscriptions

- **Start the Schema Editor and click File Configurations to open a configuration file.**

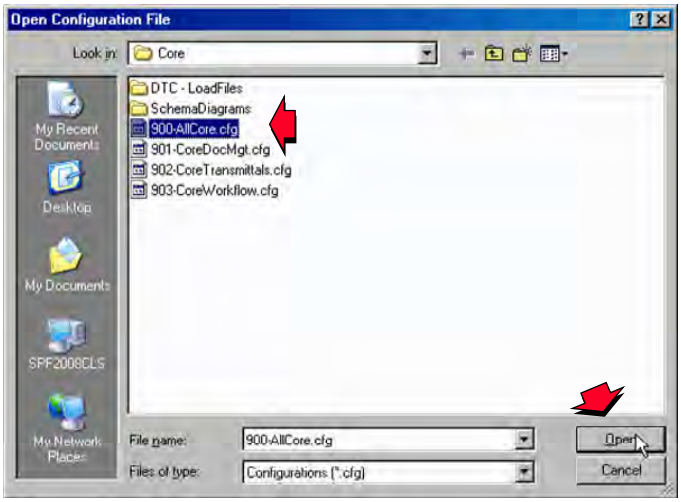


© 2008. Intergraph Corp.
All Rights Reserved.

The location for the core schema files is **\SmartPlant\Foundation\2008\Models\Core**.

Setting Up Subscriptions

- **Choose *900-AllCore.cfg* from the Core folder and click Open.**

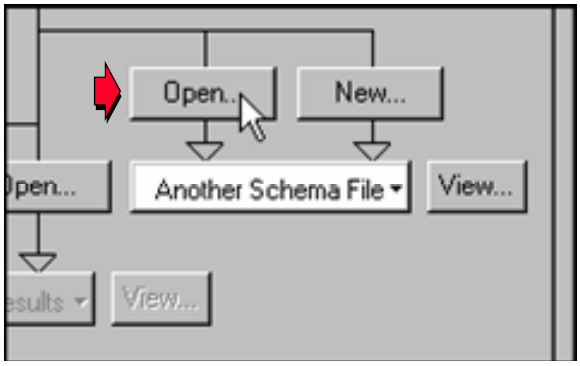


© 2008. Intergraph Corp.
All Rights Reserved.

Once the core schema files have been opened with the *900-AllCore.cfg*, the extension schema file (EISchema.xml) will need to opened.

Setting Up Subscriptions

- **Open the schema file containging the *EIPump* definitions by clicking the Open button above *Another Schema File.***

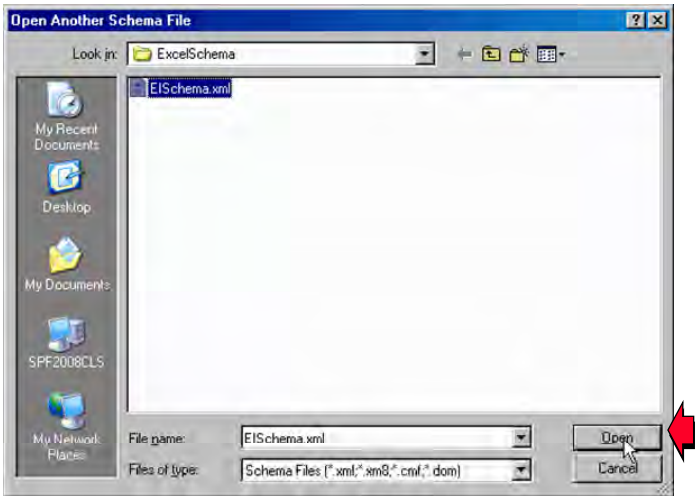


© 2008. Intergraph Corp.
All Rights Reserved.

The extension schema for this example is located in the *D:\SPF42_Training\ExcelSchema* folder.

Setting Up Subscriptions

- **Select *EISchema.xml* and click Open.**

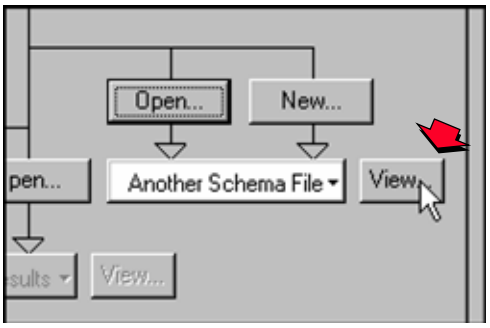


© 2008. Intergraph Corp.
All Rights Reserved.

After the needed schema files have been opened, use the **View** button to begin to make the needed modifications.

Setting Up Subscriptions

- **In the *Workflows* dialog box click the View button (next to *Another Schema File) to* View the schema.**

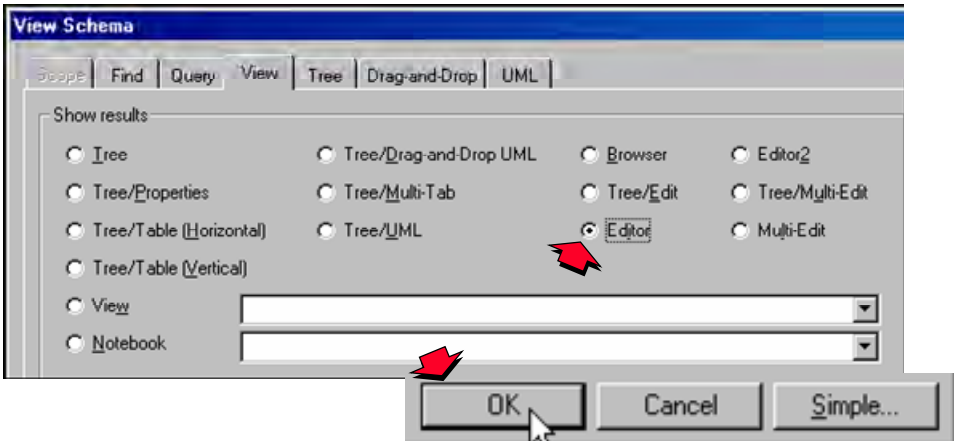


© 2008. Intergraph Corp.
All Rights Reserved.

When the *View Schema* dialog displays, choose the type of view to use for this editing session.

Setting Up Subscriptions

- **Choose the *Editor* option and click OK.**

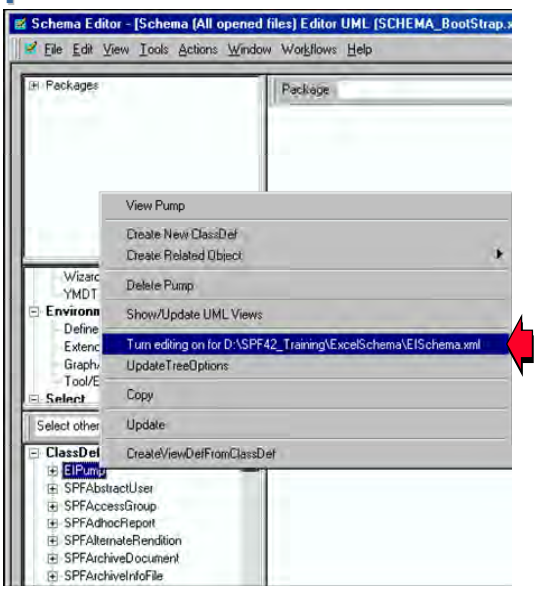


© 2008. Intergraph Corp.
All Rights Reserved.

If the Edit command is not available on the shortcut menu, turn on the editing functionality.

Setting Up Subscriptions

- **In the *ClassDefs* tree, right click on *ElPump* and from the shortcut menu, select the command to turn on editing.**

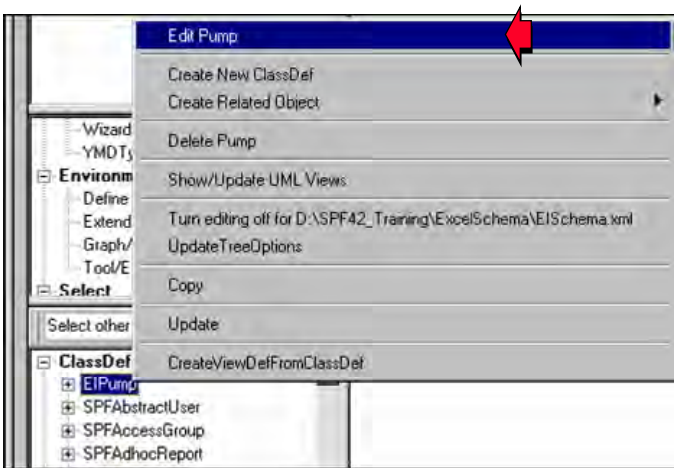


© 2008. Intergraph Corp.
All Rights Reserved.

Next, edit the schema to add the necessary interface definition.

Setting Up Subscriptions

- **Next, right click on *ElPump* and from the shortcut menu, select *Edit Pump*.**



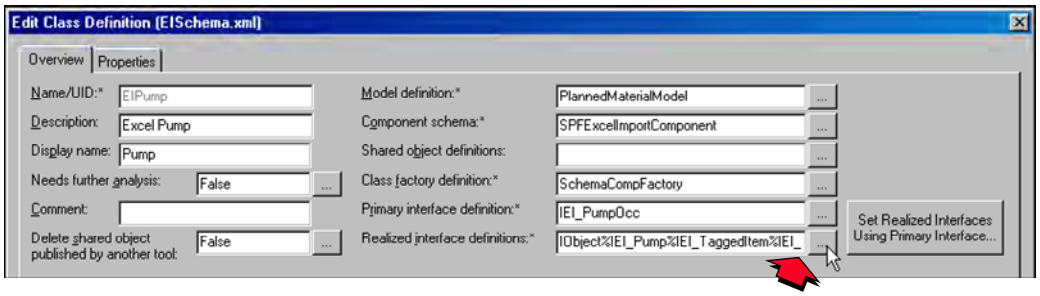
© 2008. Intergraph Corp.
All Rights Reserved.

The *Edit Class Definition* dialog for the **EIPump** class will appear. The objective is to add the **ISPFSubscribableItem** interface as a realized interface for this class.

INTERGRAPH

Setting Up Subscriptions

- **Choose the browse button to edit *Realized interface definitions.***



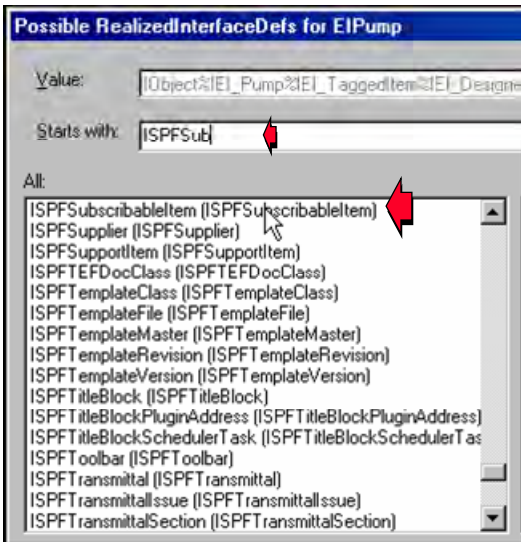
© 2008. Intergraph Corp.
All Rights Reserved.

Use the search options to locate the ISPFSubscribableItem interface.

INTERGRAPH

Setting Up Subscriptions

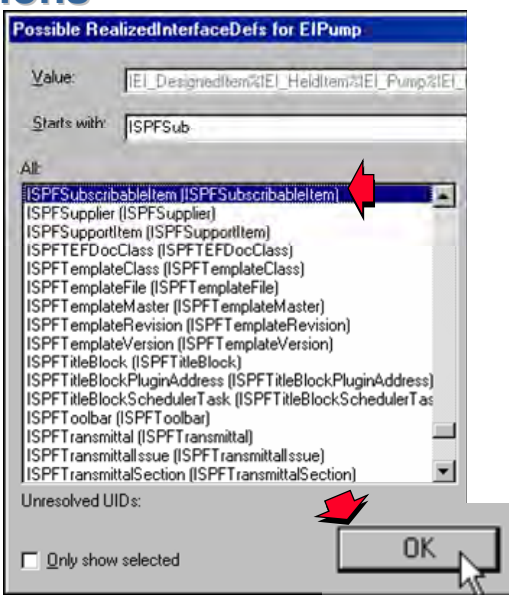
- **Key in the first few characters of the interface name to search for the *ISPFSubscribableItem* interface in the Possible *RealizedInterfaceDefs for EIPump* dialog.**



© 2008. Intergraph Corp.
All Rights Reserved.

Setting Up Subscriptions

- **Select *ISPFSubscribableItem* from the Possible *RealizedInterfaceDefs for EIPump* and click OK.**

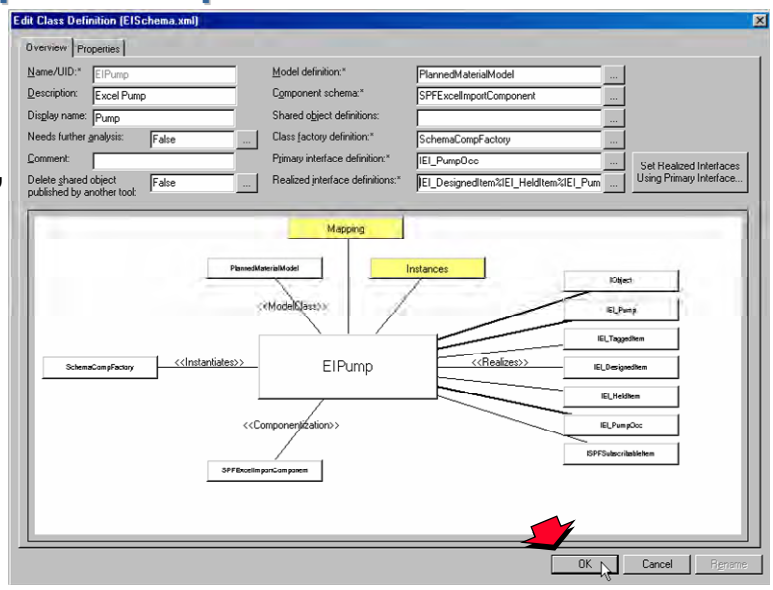


© 2008. Intergraph Corp.
All Rights Reserved.

This will add ISPFSubscribableItem as an additional interface to the **EIPump** class.

Setting Up Subscriptions

- **To relate the new Realized interface definition, click OK.**



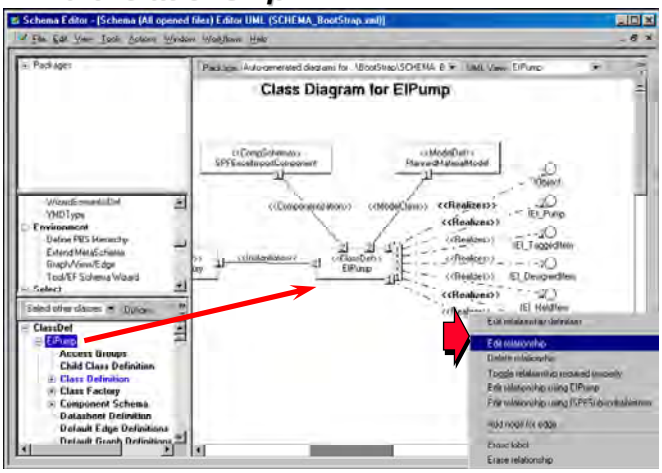
© 2008. Intergraph Corp.
All Rights Reserved.

The subscription functionality requires that the ISPFSubscribableItem interface be a **mandatory** interface.

To make this change, drag and drop the EIPump class def into the UML view window in order to create a default view.

Setting Up Subscriptions

- **Right click on the new *realizes* relationship for *ISPFSubscribableItem* and from the shortcut menu, select *Edit relationship.***

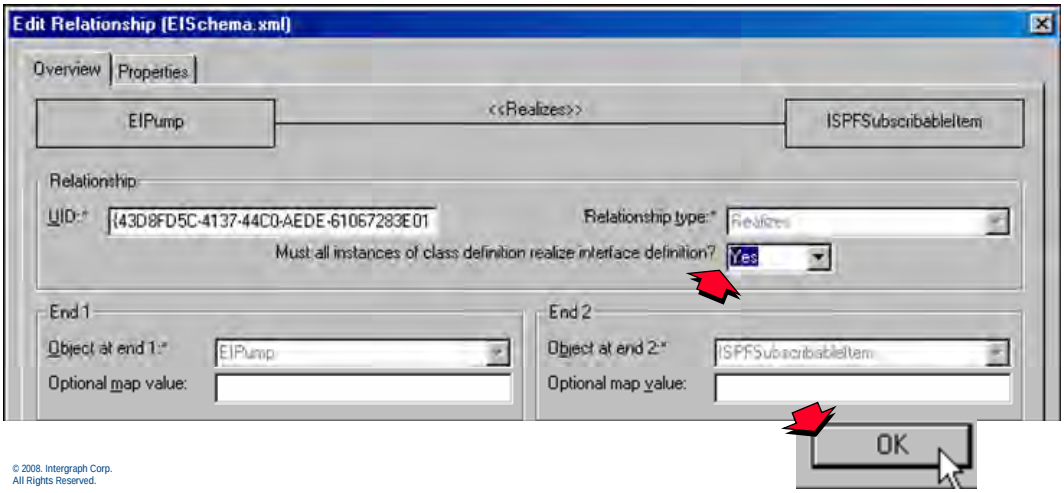


© 2008. Intergraph Corp.
All Rights Reserved.

The *Edit Relationship* dialog will appear.

Setting Up Subscriptions

- **Select *Yes* for *Must all instances of class definition realize interface definition?* and click OK.**

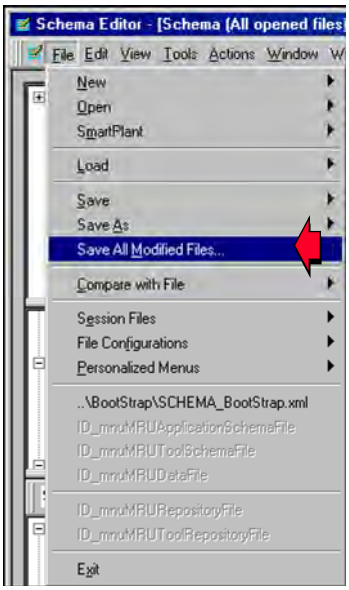


© 2008. Intergraph Corp.
All Rights Reserved.

Once the changes to the schema have been made, save the changed files.

Setting Up Subscriptions

- **From the menu, select *File > Save All Modified Files.***

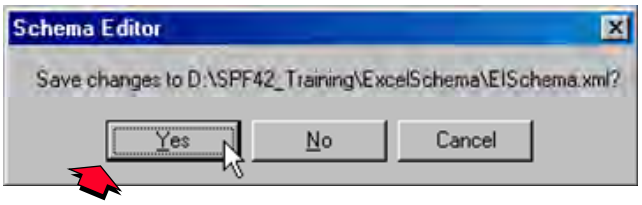


© 2008. Intergraph Corp.
All Rights Reserved.

A save dialog will appear for each file to be saved.

Setting Up Subscriptions

- **Click Yes to save changes to the "pump" schema file.**

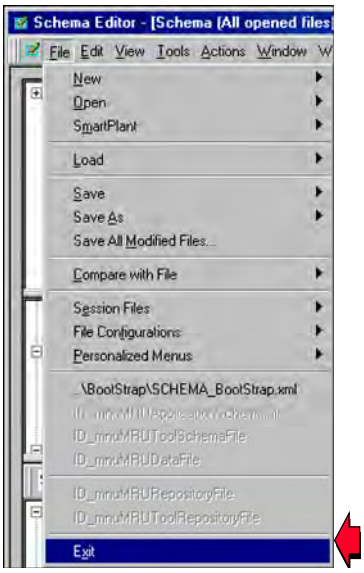


© 2008. Intergraph Corp.
All Rights Reserved.

Now that the schema changes have been made, exit from the schema editor.

Setting Up Subscriptions

- **Select *File* > *Exit* from the menu to close the schema editor.**



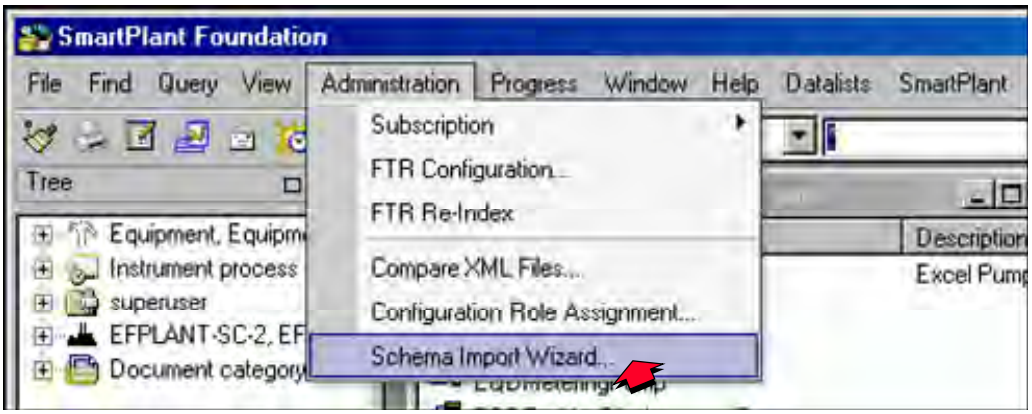
© 2008. Intergraph Corp.
All Rights Reserved.

3.5 Importing Schema Changes

For the subscription example, the changed **EIPump.xml** file will need to be loaded into the SPF database.

Importing Schema Changes

- **In the SmartPlat Foundation Desktop Client, choose *Administration > Schema Import Wizard*.**



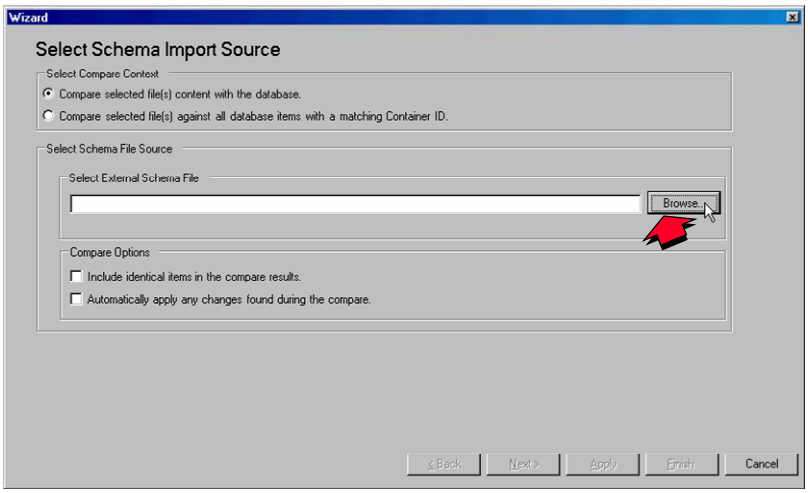
© 2008. Intergraph Corp.
All Rights Reserved.

Use the **Schema Import Wizard** from the Desktop Client to accomplish this task.

When importing a schema into SmartPlant Foundation, it is necessary to first compare the schema with the existing content of the database and then selectively load the updates.

Importing Schema Changes

- **In the *Schema Import Wizard*, click the *Browse* button to search for the schema file to be loaded.**

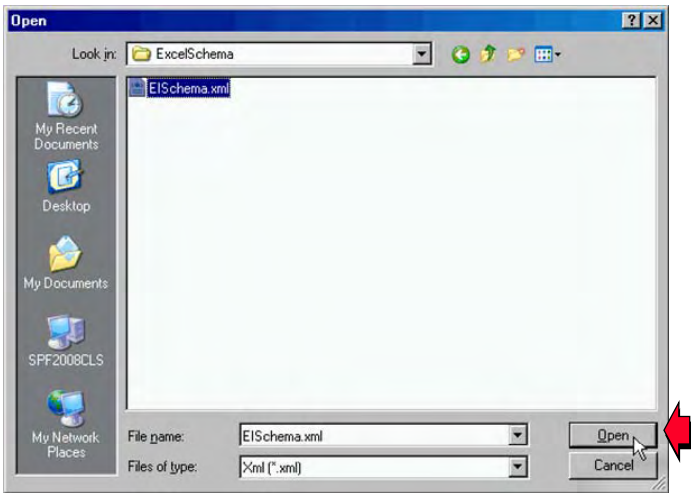


© 2008. Intergraph Corp.
All Rights Reserved.

When the browser dialog appears, select the schema file to be imported.

Importing Schema Changes

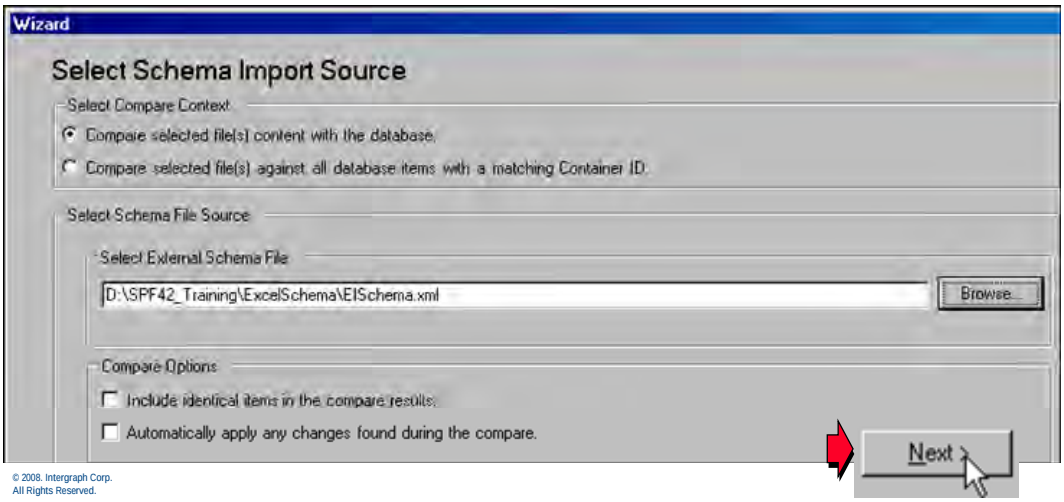
- **Select the *EISchema.xml* file and click Open.**



© 2008. Intergraph Corp.
All Rights Reserved.

Importing Schema Changes

- **Click Next to compare schema file content with the database.**

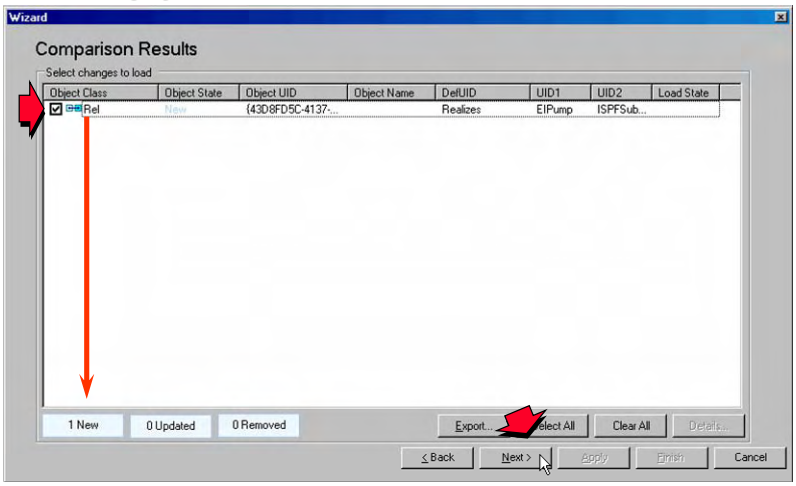


© 2008. Intergraph Corp.
All Rights Reserved.

The *Comparison Results* form will appear. To selectively load updates, enable the corresponding checkboxes next to the object to be loaded.

Importing Schema Changes

- **Select the new *Realizes* relationship to be loaded and click Next.**

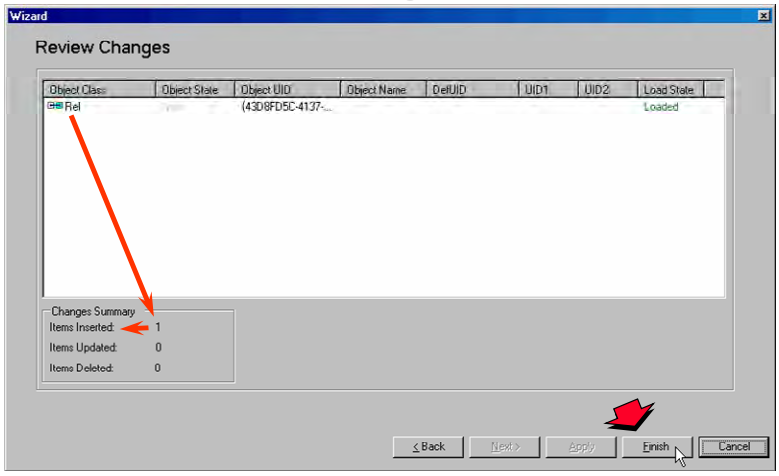


© 2008. Intergraph Corp.
All Rights Reserved.

The totals (*New*, *Updated* and *Removed* objects and relationships) are displayed at the bottom of the form.

Importing Schema Changes

- **Verify that the new *Realizes* relationship is loaded to the database and click Finish.**



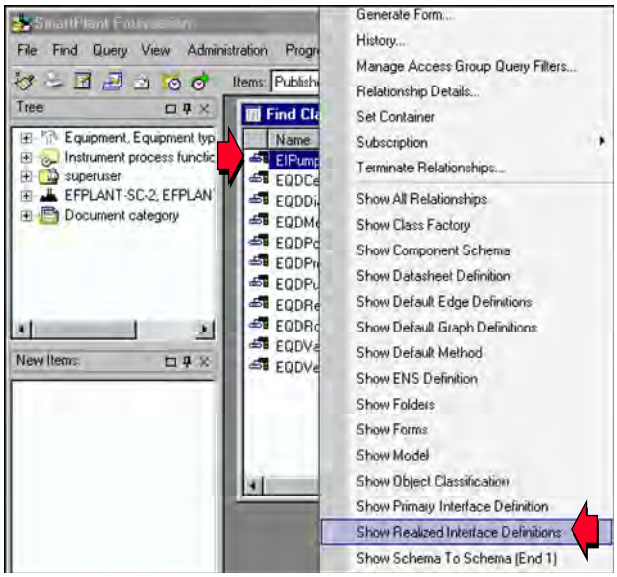
© 2008. Intergraph Corp.
All Rights Reserved.

After the update, the wizard displays the load status for each object and a summary in the *Review Changes* form.

Use the ***Find*** > ***Schema*** > ***Class Definition*** command to display the **EIPump** class definition.

Verifying the Schema Changes

- **Right click on the *EIPump* ClassDef and from the shortcut menu, select *Show Realized Interface Definitions.***

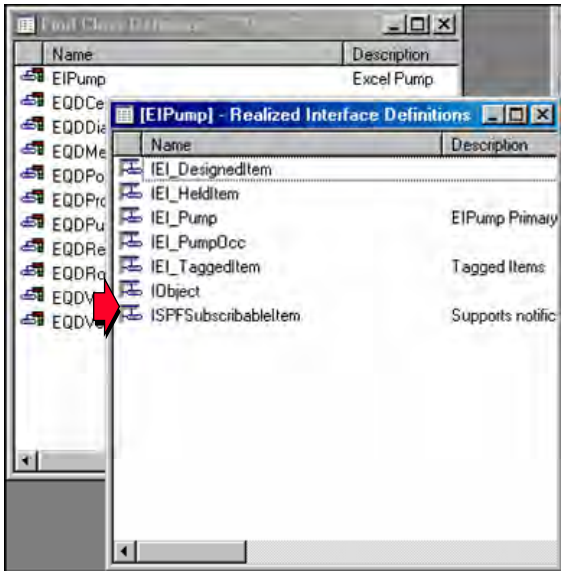


© 2008. Intergraph Corp.
All Rights Reserved.

An *EIPump - Realized Interface Definitions* list view window will appear.

Verifying the Schema Changes

- **Verify that the *ISPFSubscribableItem* is displayed as a realized interface defintion.**



© 2008. Intergraph Corp.
All Rights Reserved.

3.5.1 Creating Log Subscriptions

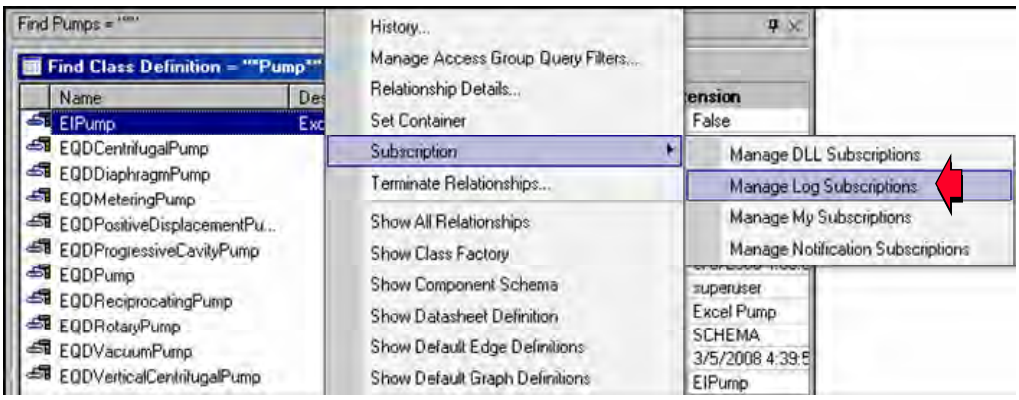
Log subscriptions send logging data to a specified file. The text that gets logged will be the text of the appropriate *Notification Definition*, which is derived in the same way as for Notification subscription. This will support text substitutions just the same as Notification subscription.

Find and display the object class in order to set up a log subscription.

INTERGRAPH

Administrative Log Subscriptions

- **Right click on the *ElPump* ClassDef and select *Subscription > Manage Log Subscriptions* from the shortcut menu.**

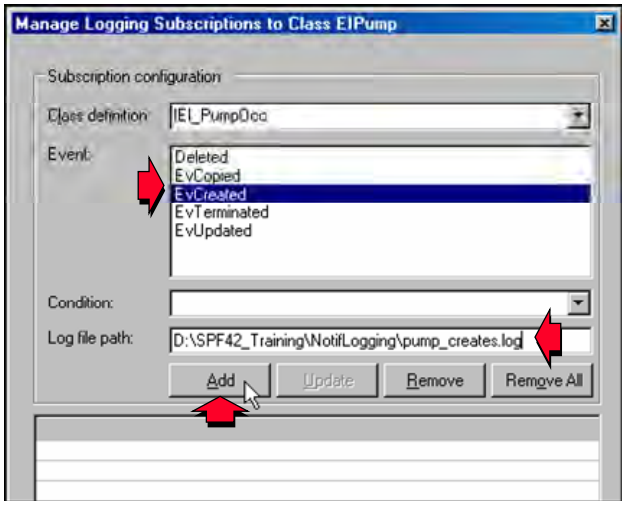


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Logging Subscriptions to Class* form will appear.

Administrative Log Subscriptions

- **Select the *EvCreated* event, enter a log file path and name then click Add.**



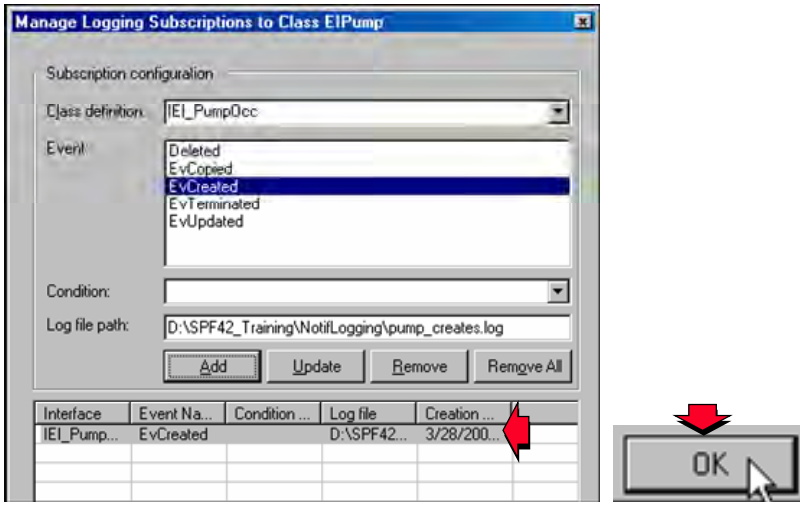
© 2008. Intergraph Corp.
All Rights Reserved.

These are the fields available in the *Manage Logging Subscriptions to Class* form:

- **Class definition** - This lists all the interface definitions for the selected class with the primary interface pre-selected.
- **Event** - Select the event that will trigger the subscription.
- **Condition** - Select any condition that will apply to the subscription.
- **Log file path** - The path to where the log file will be written, for example **D:\SPF42_Training\NotifLogging\pump_creates.log**.

Administrative Log Subscriptions

- Verify the selected event and logging information are added and click **OK**.

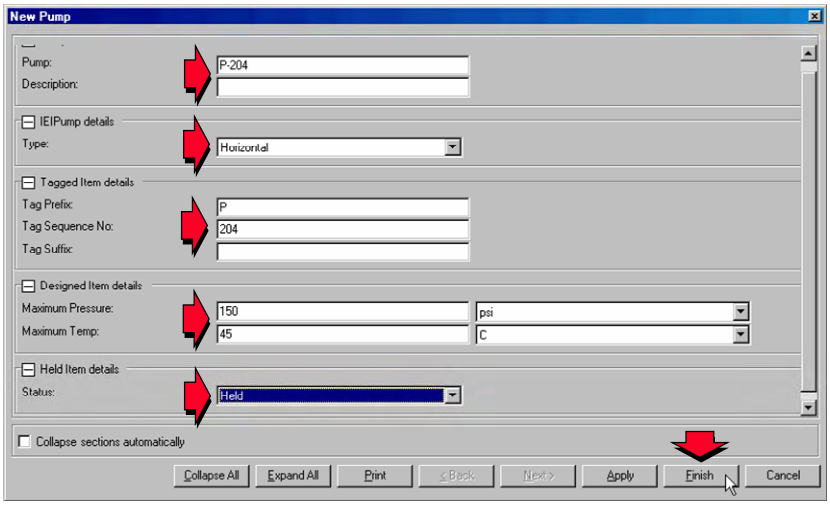


© 2008. Intergraph Corp.
All Rights Reserved.

To test the log subscription, a user (Eddie) will create a new pump instance.

Testing Log Subscriptions

- Create a pump instance by entering some property values and click **Finish**.

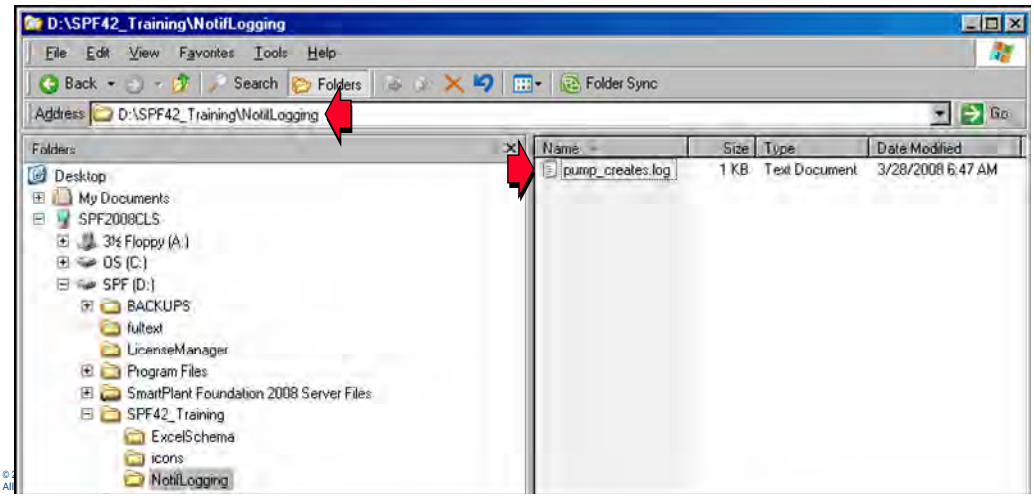


© 2008. Intergraph Corp.
All Rights Reserved.

Executing the ***File*** > ***New*** > ***Pump*** command will cause the file **pump_creates.log** to be created.

Testing Log Subscriptions

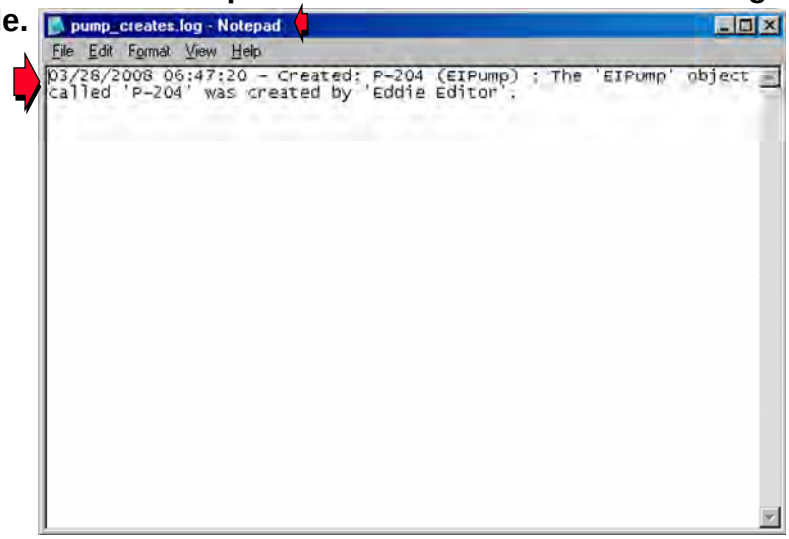
- **Use Windows Explorer to locate the log file that was created.**



The log file will contain such information as date, time, the object instance that was created, and who created the object.

Testing Log Subscriptions

- **Double-click to open and view the contents of the log file.**



© 2008. Intergraph Corp.
All Rights Reserved.

3.6 Creating Custom Notifications

Notification is the mechanism that informs the user that made the notification subscription. Notification definitions are used to configure the message that is issued to a user because of notification subscription.

When creating a notification, text substitution is supported, for example:

> 'The value for pressure (&OBJ.PRESSURE&) on &OBJ.NAME& is too high for the stated piping materials class (&OBJ. PIPINGMATERIALCLASSIFICATION&).'

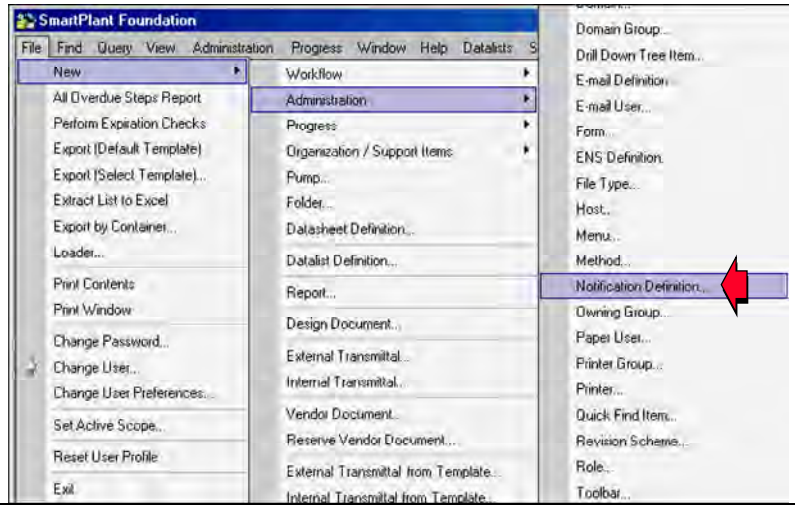
This will result in a message similar to '**The value for pressure (50psi) on P-10-5"-2001 is too high for the stated piping materials class (CM).**'

Use the following steps to configure a custom *Notification Definition*.

INTERGRAPH

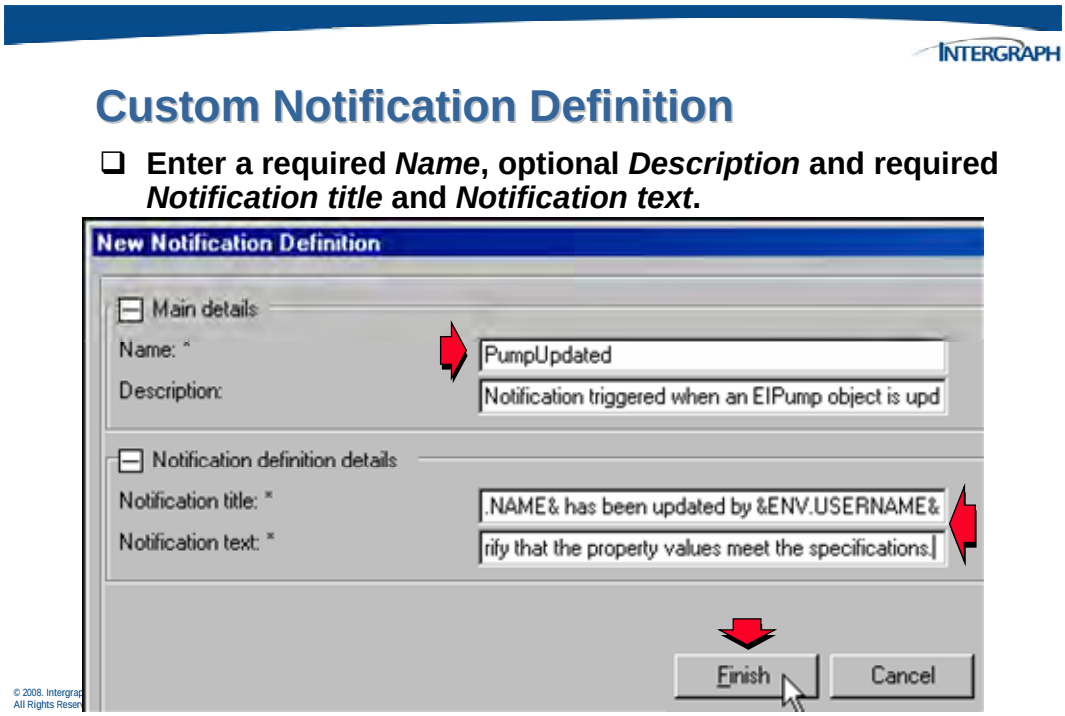
Custom Notification Definition

- **Select *File > New > Administration > Notification Definition* from the menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *New Notification Definition* form will appear.



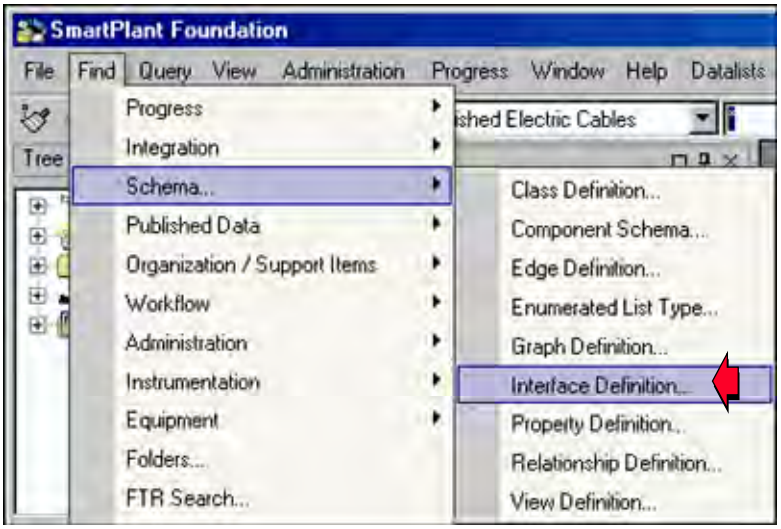
Below are the *Description*, *Notification title* and *Notification text* shown in the above example:

- **Description:** Notification triggered when an EIPump object is updated.
- **Notification Title:** &OBJ.NAME& has been updated by &ENV.USERNAME&
- **Notification text:** The Pump "&OBJ.NAME&" has been updated by user &ENV.USERNAME&. Please verify that the property values meet the specifications.

Now that the Notification Definition object has been created, it will need to be related to an object class via an interface definition.

Custom Notification Definition

- **From the menu select *Find* > *Schema* > *Interface Definition*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Interface Definition* dialog will appear.

Custom Notification Definition

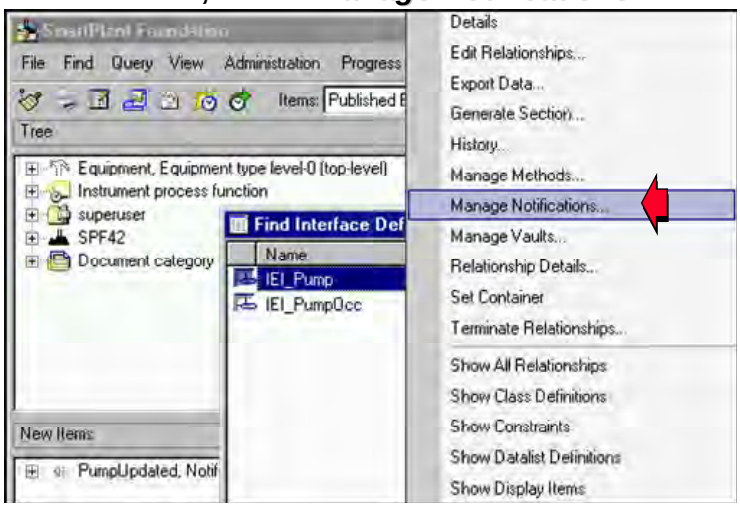
- **Enter the criteria to locate the custom ElPump realized interface definitions.**

© 2008. Intergraph Corp.
All Rights Reserved.

Next, relate an event with the interface definition.

Custom Notification Definition

- **Right click on the *IEIPump* InterfaceDef and from the shortcut menu, select *Manage Notifications.***

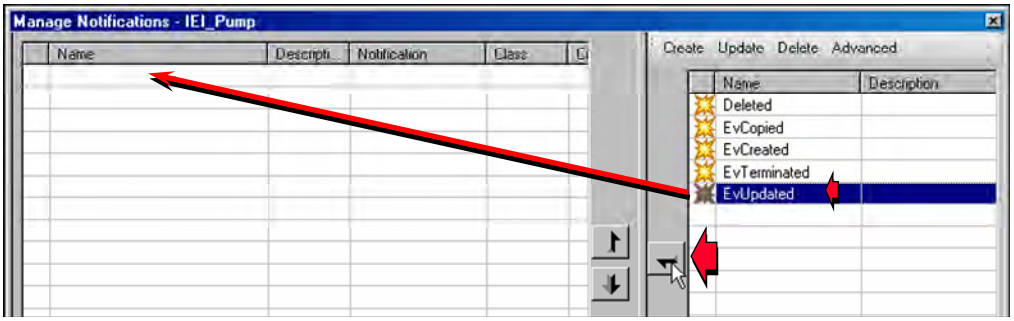


© 2008. Intergraph Corp.
All Rights Reserved.

A list of available events will be displayed in the *Manage Notifications* form.

Custom Notification Definition

- **Select the *EvUpdated* Event and click the left arrow button to relate this event to the *IEI_Pump* interface.**

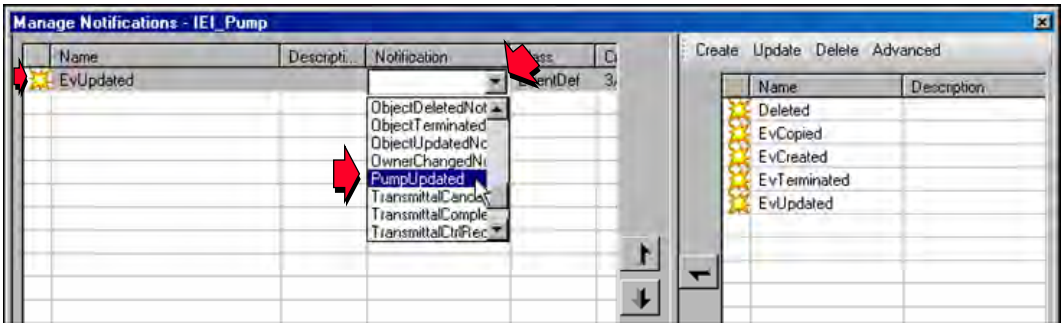


© 2008. Intergraph Corp.
All Rights Reserved.

For this update event, the custom notification will replace the default notification.

Custom Notification Definition

- **Click in the *Notification* field in order to associate the *EvUpdated* event with the new custom *Notification Definition* – PumpUpdated.**

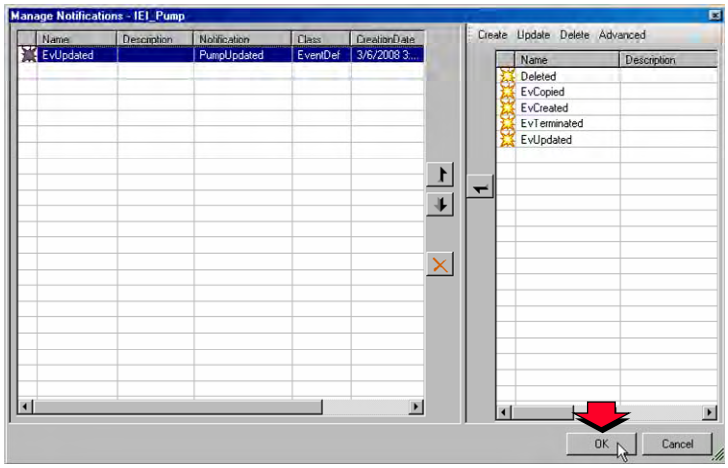


© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the steps if other events will be configured with custom notifications.

Custom Notification Definition

- **Once the custom *Notification Definition* has been associated with the update event, click OK to save these changes.**

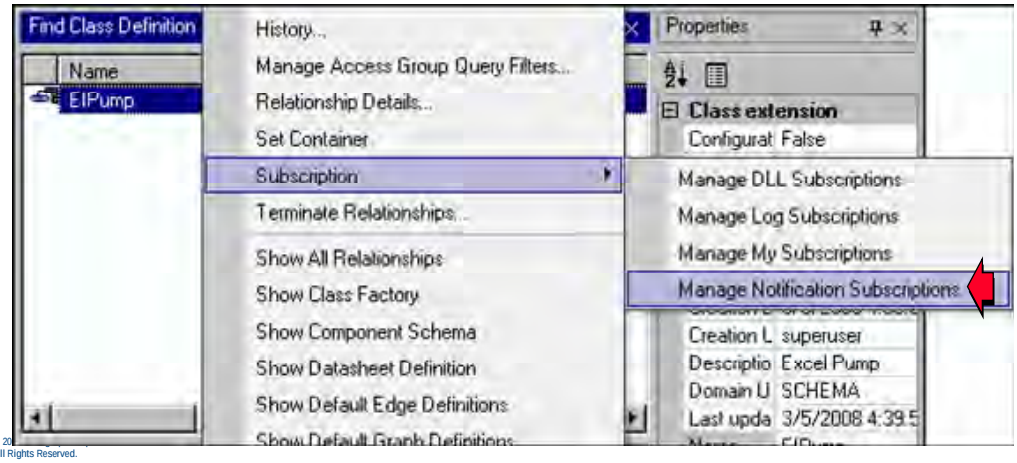


© 2008. Intergraph Corp.
All Rights Reserved.

To see the new notification, create a subscription for the pump class.

Custom Notification Definition

- **Right click on the *ElPump* ClassDef and select *Subscription > Manage Notification Subscriptions* from the shortcut menu.**

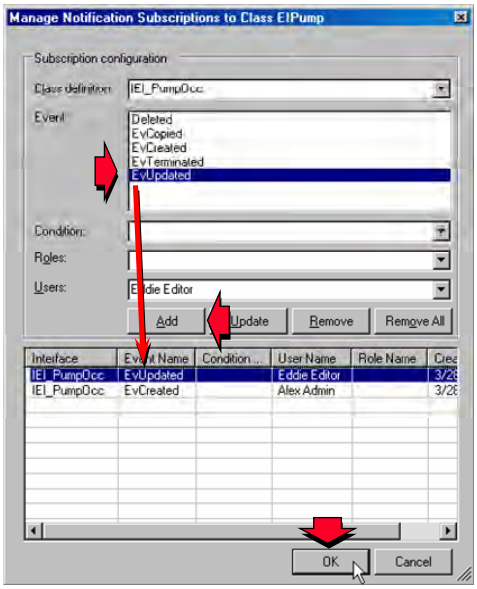


© 20
All Rights Reserved.

The update event will be subscribed to user eddie (but eddie does not have to perform the update).

Custom Notification Definition

- **Select the *EvUpdated* event, a user (Eddie Editor), click Add then click OK.**

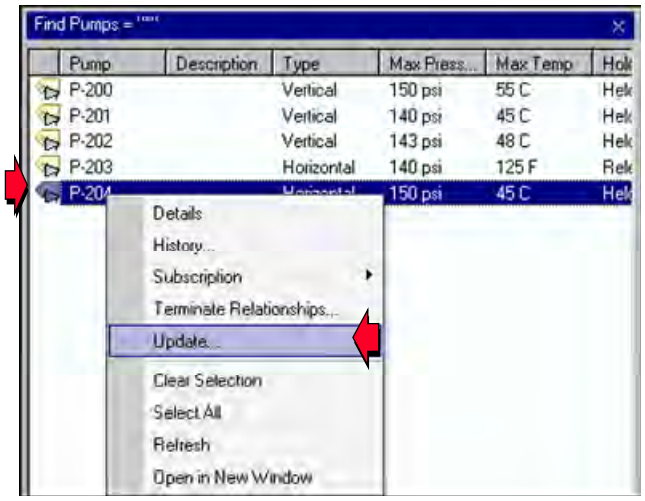


© 2008. Intergraph Corp.
All Rights Reserved.

A user (mark) will perform a ***Find*** operation to locate and display a list of pump object instances.

Testing Notification Definitions

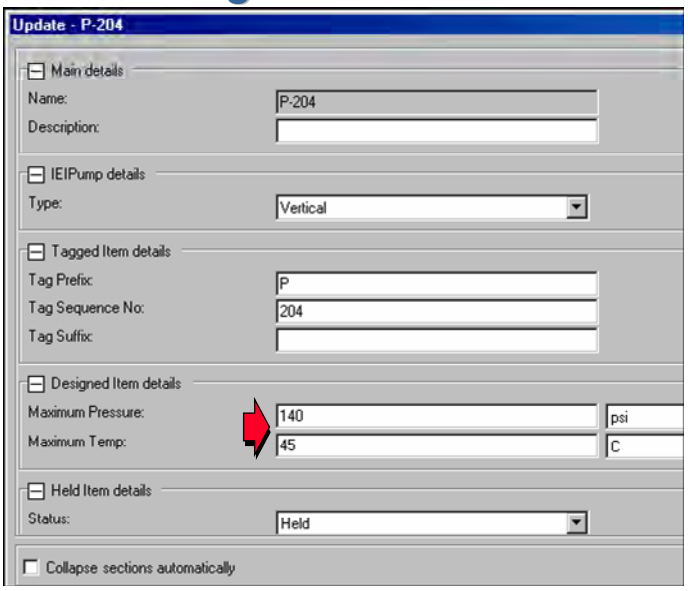
- **Right click on a pump instance and select *Update* from the shortcut menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Update* form will appear.

Testing Notification Definitions



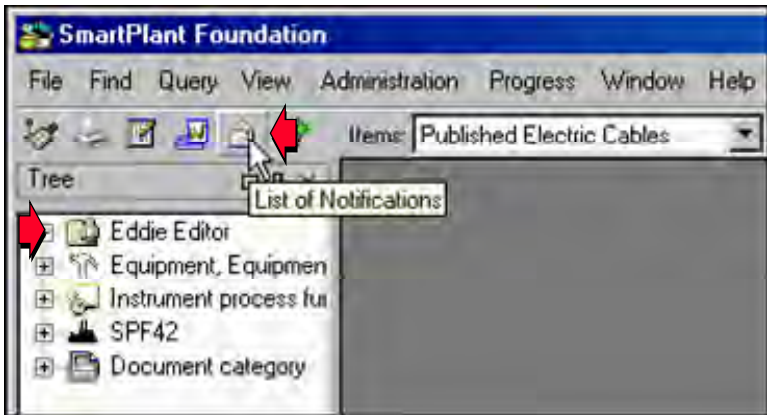
- **Update the pump by changing some property values and click Finish.**

3.6.1 Viewing Custom Notifications

To see the results of the subscription and the related custom notification definition, log in to the Desktop Client as the subscribed user.

Testing Notification Definitions

- **Select the Notification button on the toolbar to view the notifications.**

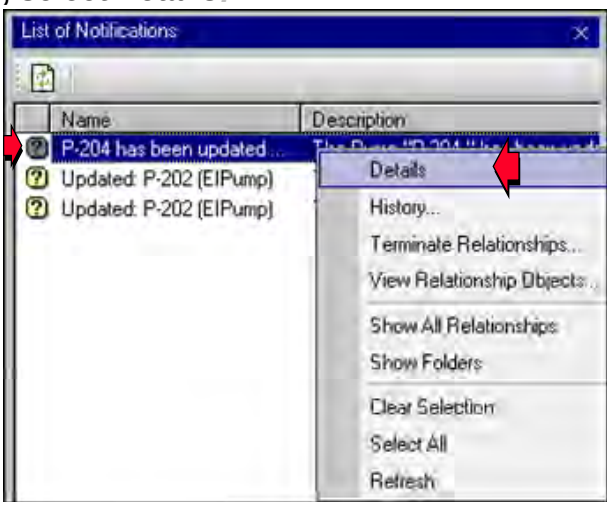


© 2008. Intergraph Corp.
All Rights Reserved.

A *List of Notifications* will be displayed in the List View.

Testing Notification Definitions

- **Right click on the notification and from the shortcut menu, select *Details*.**

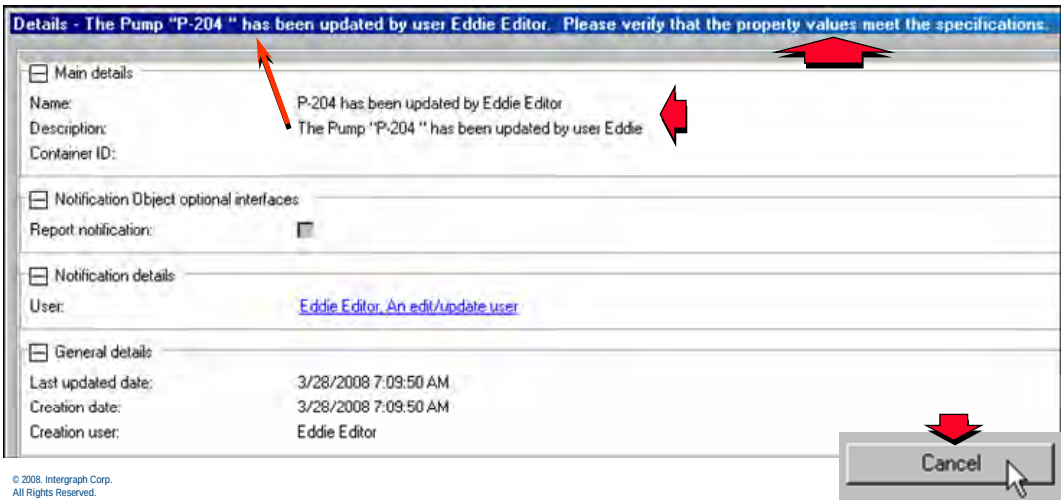


© 2008. Intergraph Corp.
All Rights Reserved.

A *Details* form will display showing the information from the custom notification.

Testing Notification Definitions

- **View the information for the *custom* notification then click Cancel to close the form.**



© 2008. Intergraph Corp.
All Rights Reserved.

3.7 Activity 1 – Subscriptions and Notifications

Complete **Chapter 3 - Activity 1** in the SPF Configuration and Administration II activity workbook.

3.8 SmartPlant Foundation Scheduler

SmartPlant Foundation Scheduler

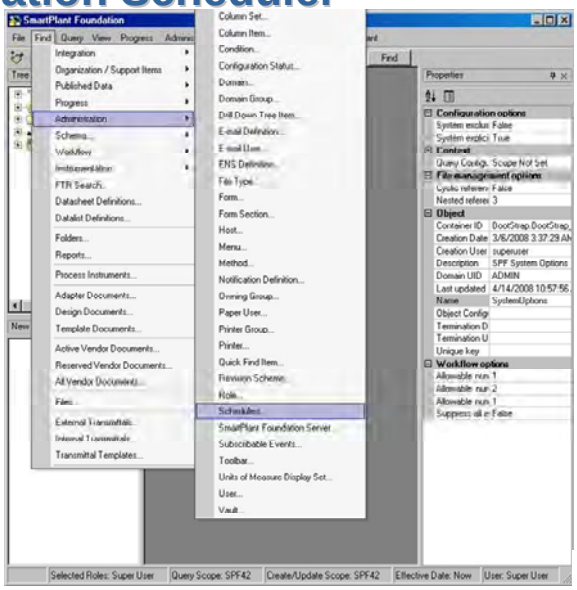
- The SmartPlant Foundation Scheduler is a way to queue up and process asynchronous tasks at a central location with queue management and status display.
- The interface for managing the Scheduler is available only from within the Desktop Client.
- Users in administrative roles can start, stop, or pause the scheduler to change the scheduler that is being viewed, changing from one database to another.
- Scheduler information display in the list view window can be filtered and sorted, as necessary.

© 2008. Intergraph Corp.
All Rights Reserved.

The command to access the scheduler viewer is illustrated below:

SmartPlant Foundation Scheduler

- To find the schedulers that you have access to view, click ***Find > Administration > Schedulers***.
- From the ***Find Schedulers*** dialog that appears, provide a name for searching or click ***OK*** to find all schedulers.

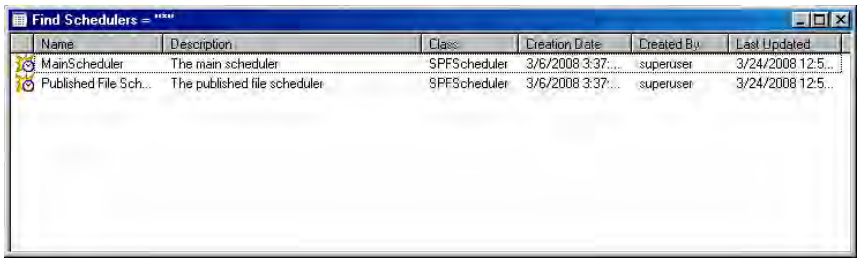


© 2008. Intergraph Corp.
All Rights Reserved.

The results of the quick find appear in a List View window.

SmartPlant Foundation Scheduler

- **The List View window displays a list of scheduler that match your search criteria.**

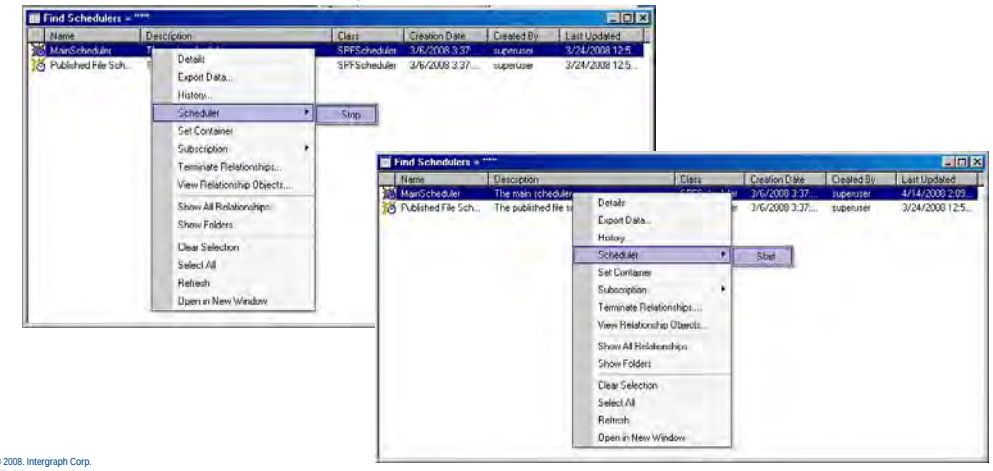


© 2008. Intergraph Corp.
All Rights Reserved.

Users with the appropriate privileges based on their roles can start and stop scheduler tasks as necessary using the shortcut menu for the task.

SmartPlant Foundation Scheduler

- **From this display, an administrator can stop or start a scheduler using the shortcut menu.**

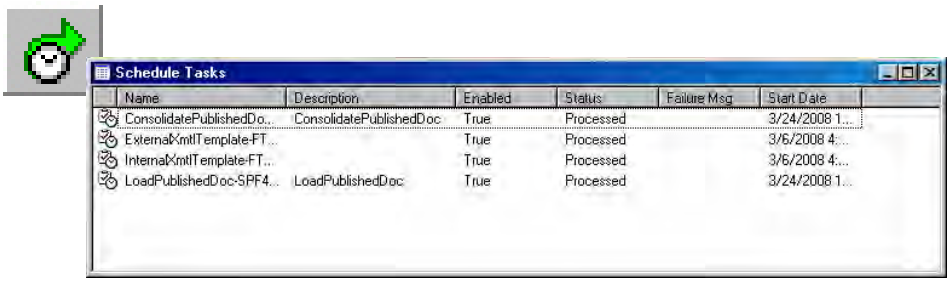


© 2008. Intergraph Corp.
All Rights Reserved.

The ***Query for Scheduled Tasks*** toolbar button provides direct access to a list of scheduled tasks.

SmartPlant Foundation Scheduler

- **The *Query for Scheduled Tasks* button on the toolbar allows you to query the scheduler using a query form and return only tasks that meet the criteria you provided.**

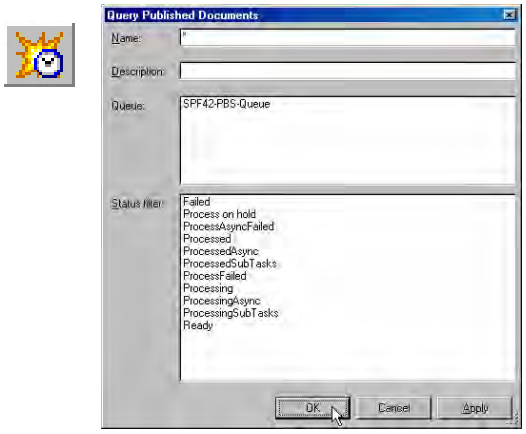


© 2008. Intergraph Corp.
All Rights Reserved.

The ***Documents in Queue*** toolbar button opens a query form so you can query for specific documents or types of documents waiting in the queue.

SmartPlant Foundation Scheduler

- **The *Documents in Queue* button on the toolbar allows you to query the documents that are currently waiting in the scheduler queue using the dialog box below:**

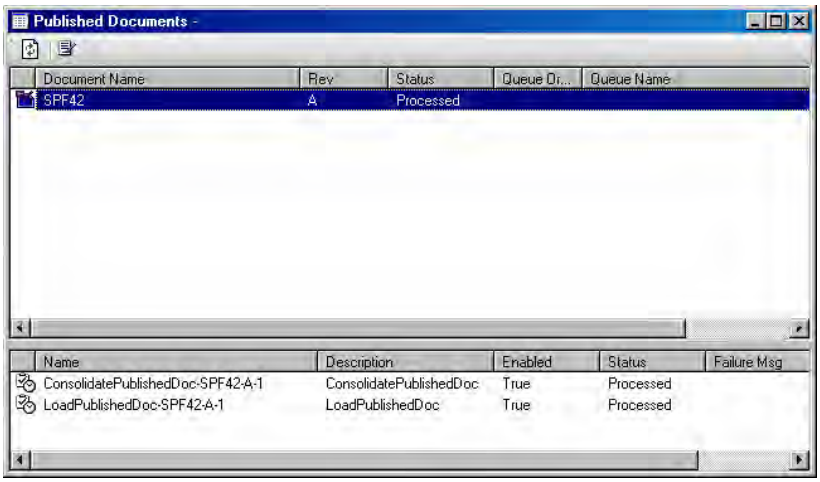


© 2008. Intergraph Corp.
All Rights Reserved.

The results of the query appear in the ***Published Document*** form illustrated below. Information about the status of the document appears in the lower pane of the window.

SmartPlant Foundation Scheduler

- **The results of your query are displayed in the *Published Documents* window.**



© 2008. Intergraph Corp.
All Rights Reserved.

3.9 Configuring SmartPlant Foundation E-mail

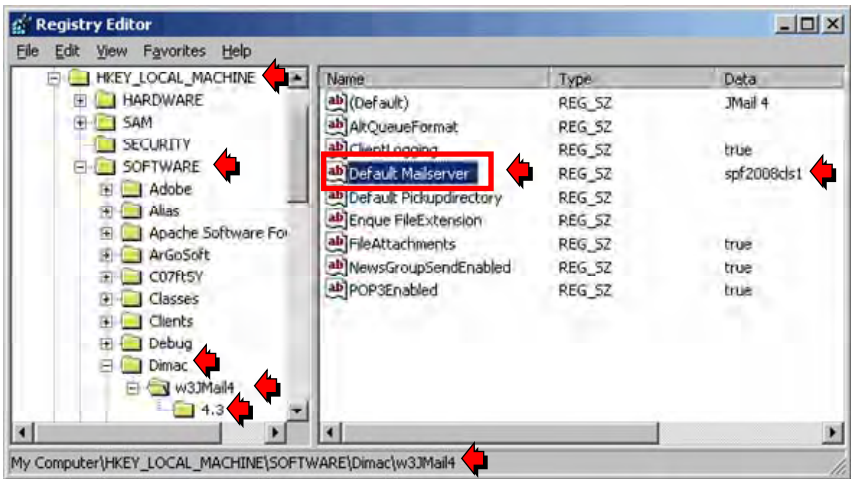
SmartPlant Foundation can be configured to use either the mail component JMAIL from Dimac Corporation (http://www.dimac.com) or Collaboration Data Objects for Windows NT Server (CDONTS or CDO).

If using JMAIL, then set a *default* mail server:

INTERGRAPH

Configuring SPF Mail

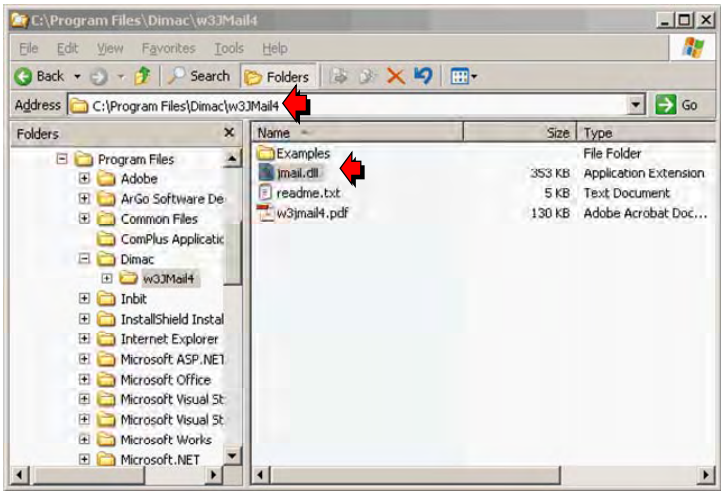
- **Edit the registry key HKEY_LOCAL_MACHINE\SOFTWARE\Dimac\w3JMail4\4.3\Default Mailserver and set it to the host name of the local mail server.**



© 2008. Intergraph Corp.
All Rights Reserved.

Configuring SPF Mail

JMAIL requires the jmail.dll be available in the folder path registered by the dll registration server (regsvr32).

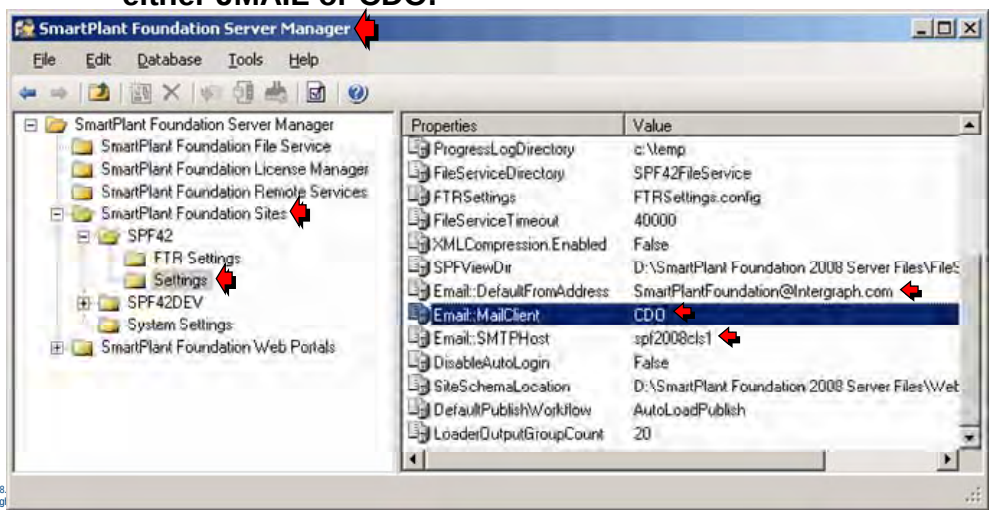


© 2008. Intergraph Corp.
All Rights Reserved.

To set the mail transport method for SmartPlant Foundation, change the *Email::MailClient* information using Server Manager.

Configuring SPF Mail

- **Use Server Manager to configure the mail client method, either JMAIL or CDO.**



© 2008.
All Rig

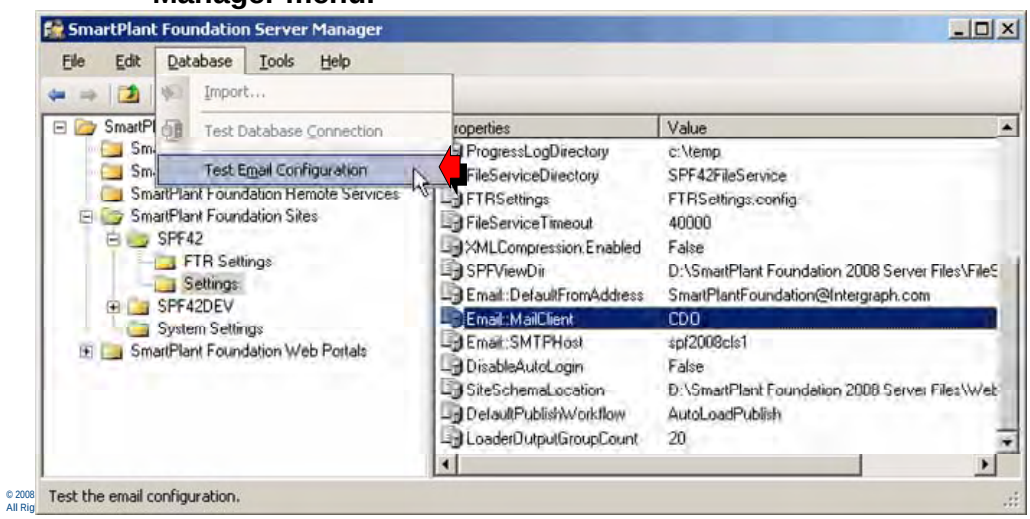
Values are JMAIL for JMAIL and CDO for CDONTS (CDO is the default value).

3.9.1 Testing the E-mail Configuration

A test harness is supplied to test the SmartPlant Foundation e-mail interface. To run the test harness to test e-mail in SmartPlant Foundation, do the following:

Testing the Email Configuration

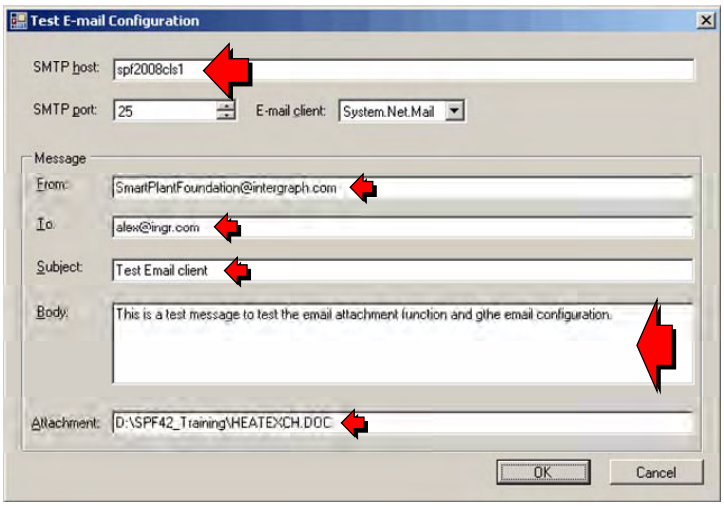
- **Select *Database > Test Email Configuration* from the Server Manager menu.**



Note: The Simple Mail Transport Protocol (SMTP) must be installed and running on the mailserver.

Testing the Email Configuration

- **Enter the necessary information to send a test email.**



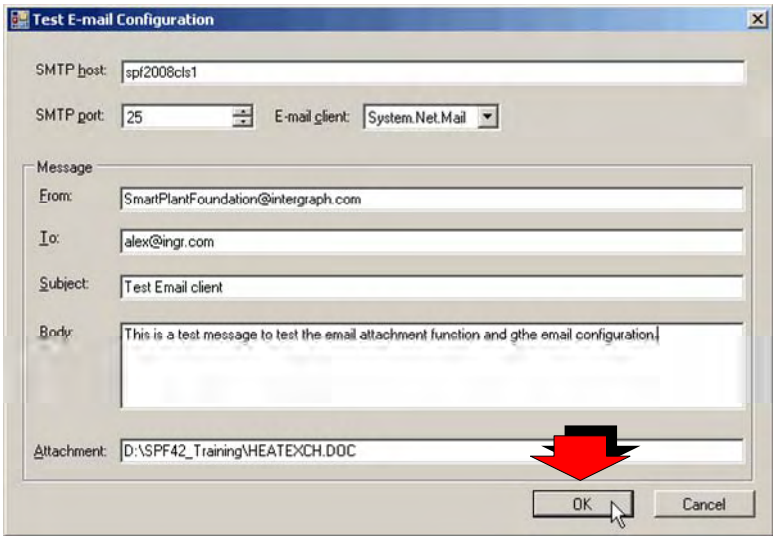
© 2008. Intergraph Corp.
All Rights Reserved.

Set the following fields on the *Test E-mail Configuration* form:

- **SMTP host –** contains the name of the mail server host
- **From –** contains the email address of the sender for the test email
- **To –** contains a valid email address of the recipient for the test email
- **Subject –** contains the subject line for the test email
- **Body –** contains a message body for the test email
- **Attachment –** key in a full path/file name to test the file attachment for the test email

Testing the Email Configuration

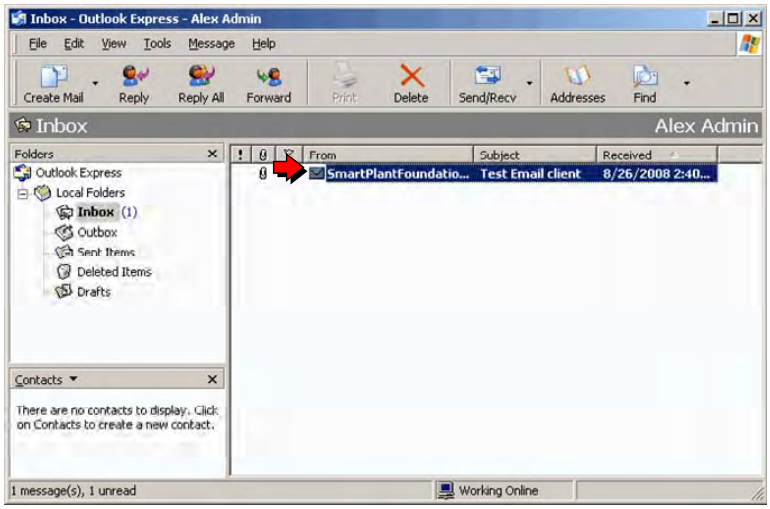
- Click **OK** to send a test email.



© 2008. Intergraph Corp.
All Rights Reserved.

Testing the Email Configuration

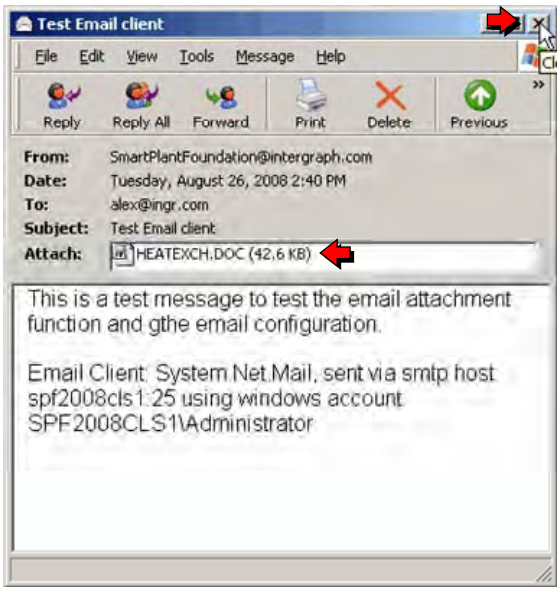
- Open the email client to verify that the test message was sent.



© 2008. Intergraph Corp.
All Rights Reserved.

Testing the Email Configuration

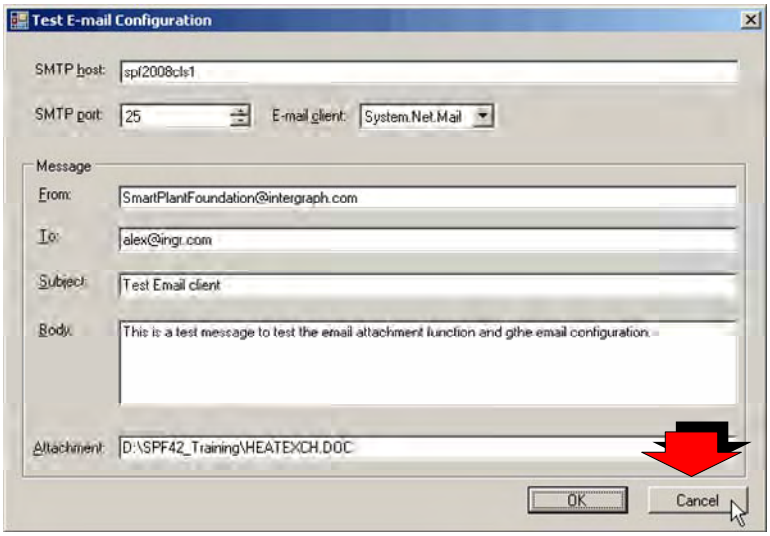
- View the test message to verify the contents and the attachment.



© 2008. Intergraph Corp.
All Rights Reserved.

Testing the Email Configuration

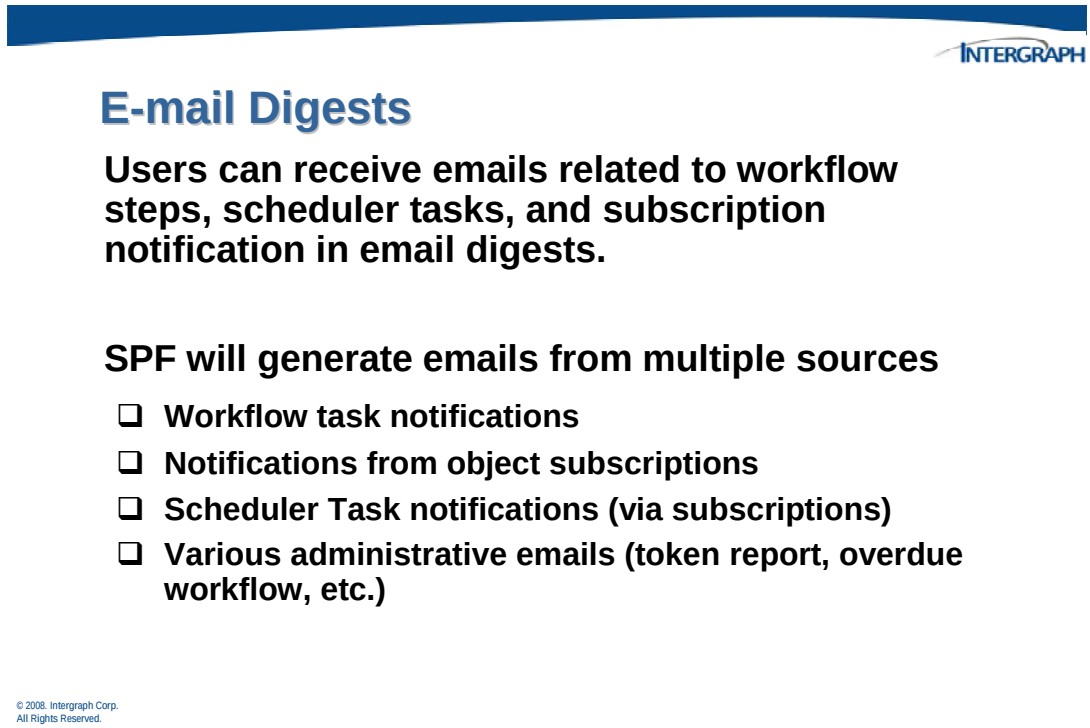
- Click **Cancel** to close the Test E-mail Configuration form.



© 2008. Intergraph Corp.
All Rights Reserved.

3.10 Configuring E-mail Digests

Users can now receive e-mails related to workflow steps, SmartPlant notifications, and subscription notification in e-mail digests, instead of as individual e-mails.



E-mail digests can be configured to send notifications to users in batches, rather than one-at-a-time.

E-mail Digests

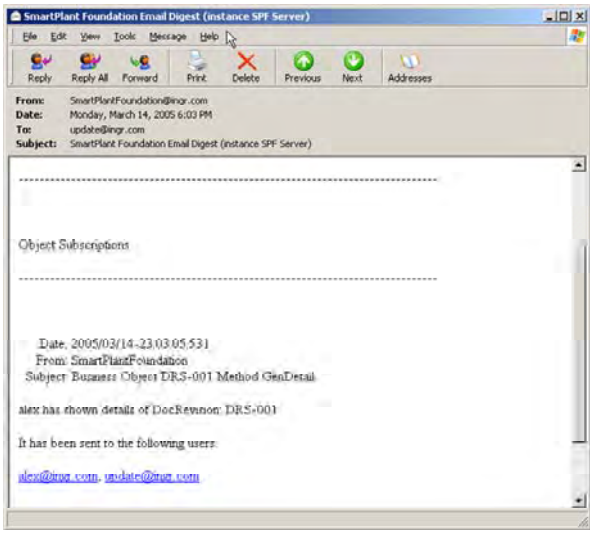
- E-mail digests are a collection of notification messages from SmartPlant Foundation that are saved up for a period of time and then sent to the recipient as a collection rather than individually.
- Users have the ability to configure if they want to use e-mail digests. The fields for configuring this are found on the *General* tab of the *Change User Preferences* dialog box.
- To customize the From address for the e-mail digest, you can edit the *Email::DefautFromAddress* property in the Settings for the applicable site in Server Manager.

© 2008. Intergraph Corp.
All Rights Reserved.

The following is an example of a Email Digest which is sent to the user.

E-mail Digests

- An email digest report appears like this example



© 2008. Intergraph Corp.
All Rights Reserved.

3.10.1 Email Digest Setup

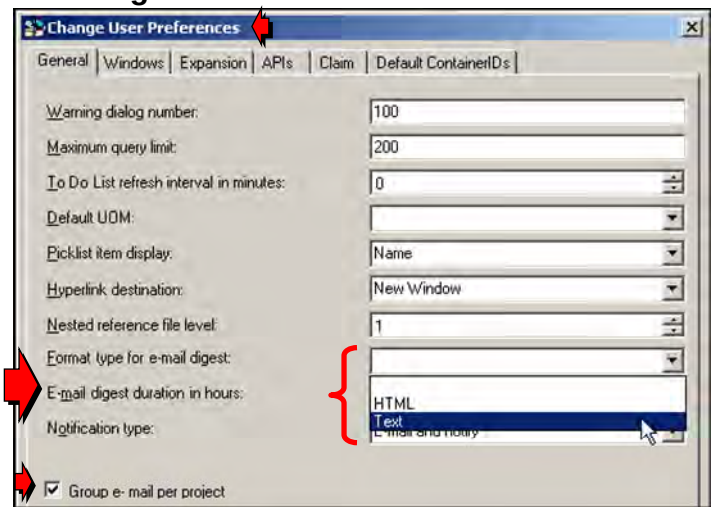
The ability to use email digests are configured by a user using the *Change User Preferences* dialog box.

Under the *General* tab, there are fields toward the bottom of the dialog that configure the **E-mail Digest** for that user.

INTERGRAPH

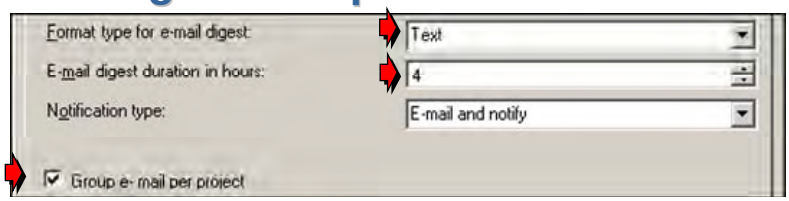
E-mail Digest Setup

- **A user preference can be set in the Desktop Client to receive digests instead of individual e-mail messages.**



© 2008. Intergraph Corp.
All Rights Reserved.

E-mail Digest Setup



- **Format type for e-mail digest – allows you to specify the format of email digests, either HTML or Text.**
- **E-mail digest duration in hours – allows you to set the interval (in hours) at which the system generates and sends email digests to the user**
- **Group e-mail per project – allows you to select whether the user receives emails regarding workflow tasks, subscriptions, and scheduler task notifications as individual emails or grouped into an email digest per project.**

© 2008. Intergraph Corp.
All Rights Reserved.

3.11 Custom Calendars

The software comes with a default, system calendar that specifies working days and non-working days for the purpose of tracking dates for workflows.

Custom Calendars

- The system uses defined calendars to keep track of working versus non-working days.
- This functionality is especially necessary when configuring workflows.
- The following steps show how you can create a new, custom calendar that may be used in place of the default system calendar that is delivered with the system.
- The delivered system calendar cannot be edited, only replaced.
- Calendars will be addressed further in the section on Workflows.

© 2008. Intergraph Corp.
All Rights Reserved.

You can create new calendars that you can customize yourself using the command illustrated below.

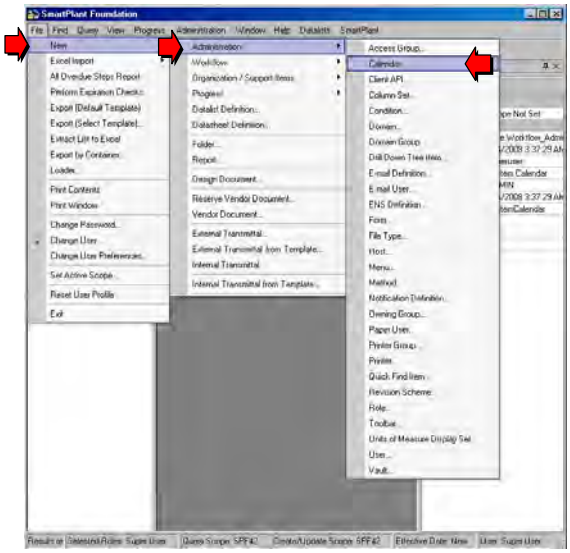
Note:

- ❑ The default calendar delivered with the system cannot be edited.

INTERGRAPH

Custom Calendars

- ❑ **To create a custom calendar, click *File > New > Administration > Calendar*.**

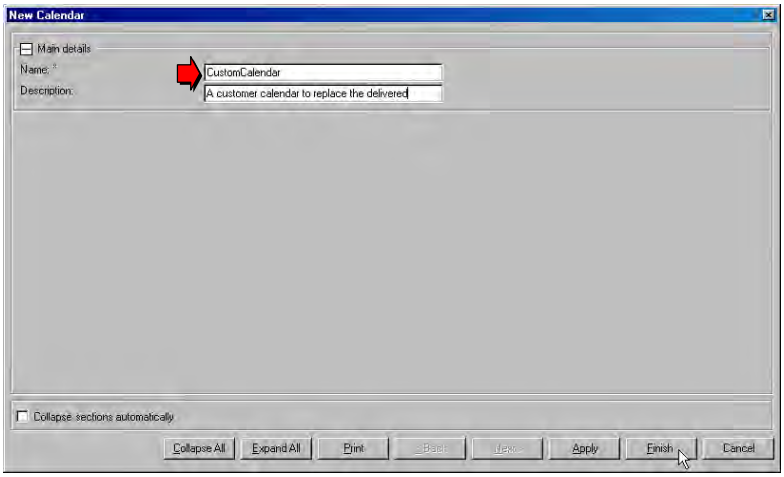


© 2008. Intergraph Corp.
All Rights Reserved.

Provide a name for the new calendar and a description, if you wish.

Custom Calendars

- **Provide a name for the new calendar on the *New Calendar* form that appears.**

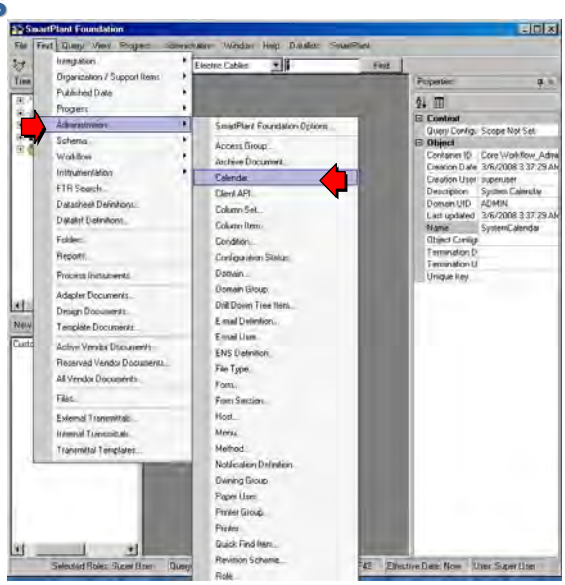


© 2008. Intergraph Corp.
All Rights Reserved.

Once you have created a new calendar, you can find it and edit it.

Custom Calendars

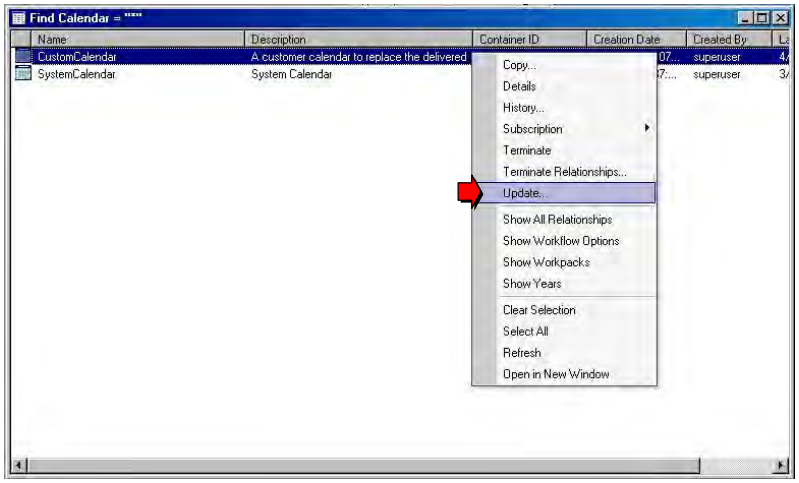
- **Once you have created a customer calendar, you can find it and configure it.**
- **Use the *Find > Administration > Calendar* command to find a list of the existing calendars.**
- **Click *OK* on the quick find dialog to find all calendars in the system or search for a specific one.**



© 2008. Intergraph Corp.
All Rights Reserved.

Custom Calendars

- **Right-click on the new calendar and use the *Update* command to open a form where you can configure it.**

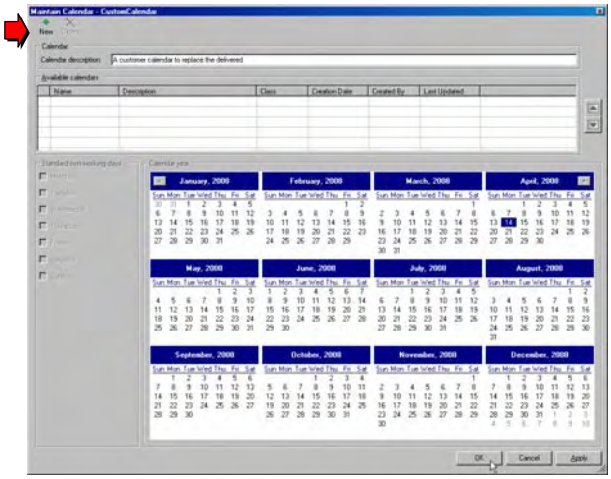


© 2008. Intergraph Corp.
All Rights Reserved.

First, click the ***New*** button to open the ***New Calendar Year*** dialog box.

Custom Calendars

- **On the *Maintain Calendar* form, click the *New* button to create a calendar for a specific year.**

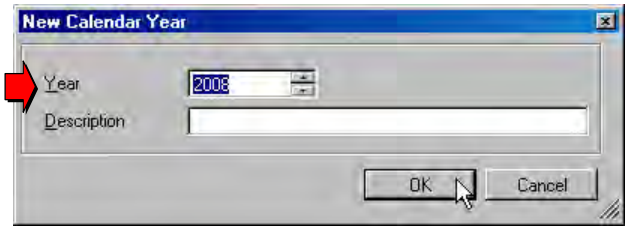


© 2008. Intergraph Corp.
All Rights Reserved.

Choose the calendar year you want to configure.

Custom Calendars

- **Choose the calendar year you want to configure, and click *OK*.**



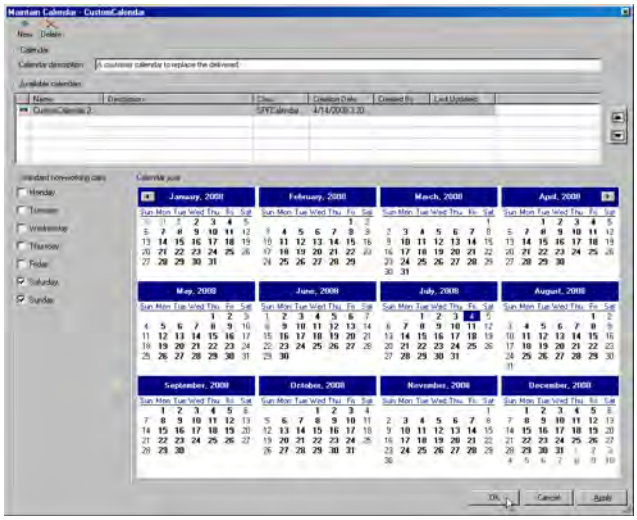
© 2008. Intergraph Corp.
All Rights Reserved.

Indicate any days of the week that should not be considered working days by activating the check box beside that name. You will notice that those days on the calendar will no longer be bold, as they are no longer working days.

INTERGRAPH

Custom Calendars

- **Configure the calendar by choose what days of the week are non-working days. Days with check boxes activated are not-bolded in the display.**
- **Click on any holidays that should not count as working days, and those will also become non-bolded to indicate that they are non-working days.**



© 2008. Intergraph Corp.
All Rights Reserved.

You may also configure the calendar further by clicking on any specific dates on the calendar that will not be working days. Those dates, too, will become non-bolded.

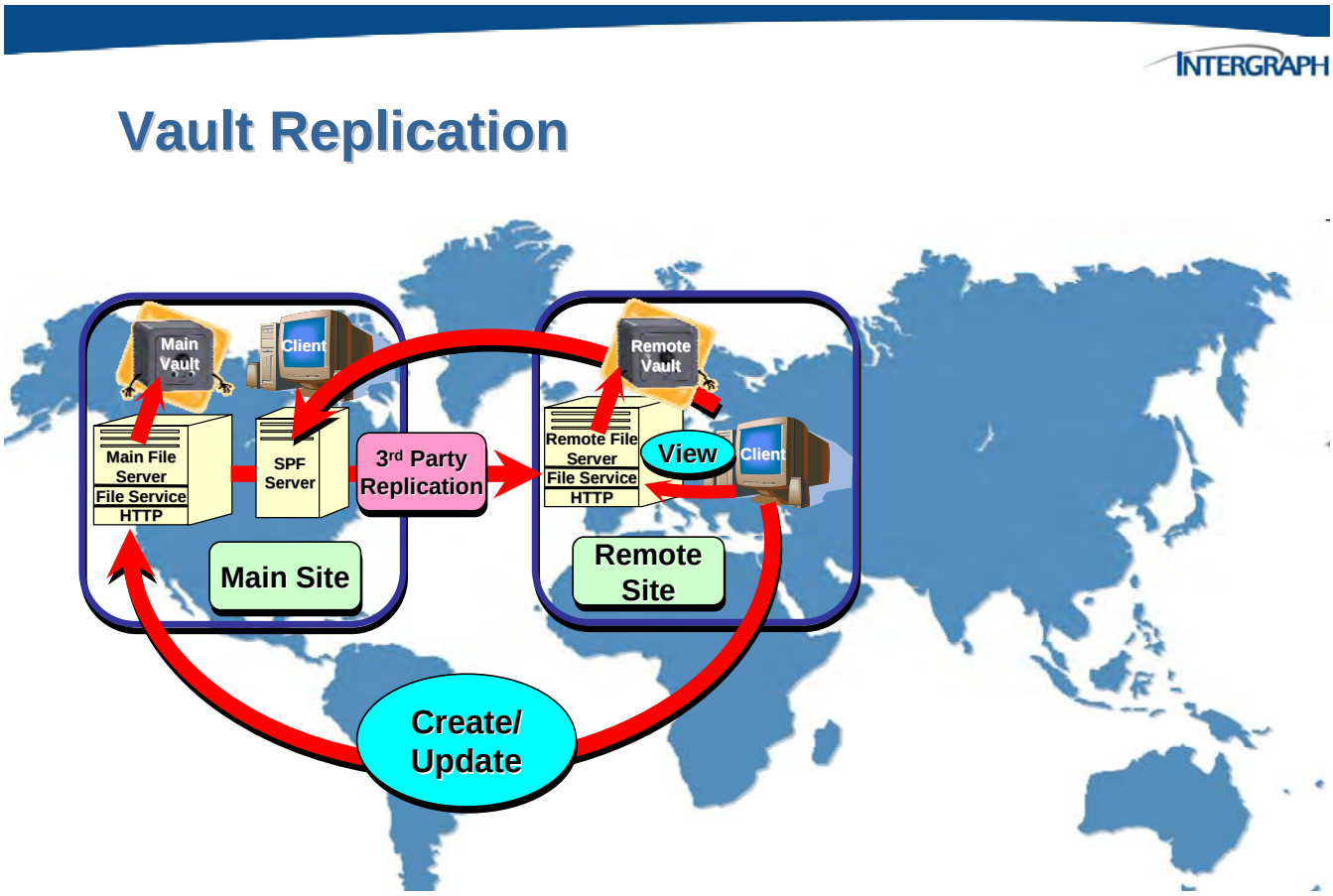
3.12 Introduction to Vault Replication

With the introduction of direct file transfer with SPF 3.6.2 it now also possible to configure file (vault) replication.

The following diagram shows how a remote SmartPlant Foundation Client access the local replicated vault for view functions.

All Create/Update functions go back to the "**Main File Server**". Viewing operations can access the replicated vault.

Note: The replication must replicate the files to the same physical location as the source vault as all the replicated vaults use the same vault information and hence need to be located in the same location.



3.12.1 Setting up a standalone File Server

The pre-requisites for setting up a local file server require the following

- .Net Framework 2.0
- Microsoft XML

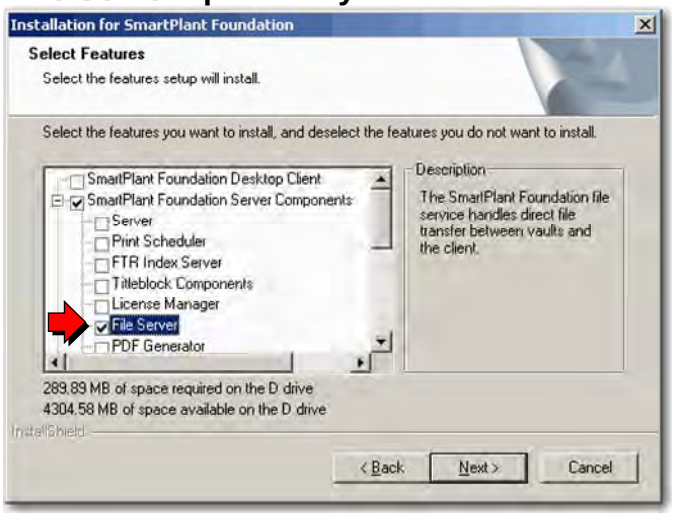
Optional

- SmartPlant Markup 3.8 (required for reading reference file headers)

INTERGRAPH

Vault Replication

- **Execute the SmartPlant Foundation installation and select the File Server option only.**



© 2005. Intergraph Corp.
All Rights Reserved.

3.12.2 Setting up Vaults for Replication

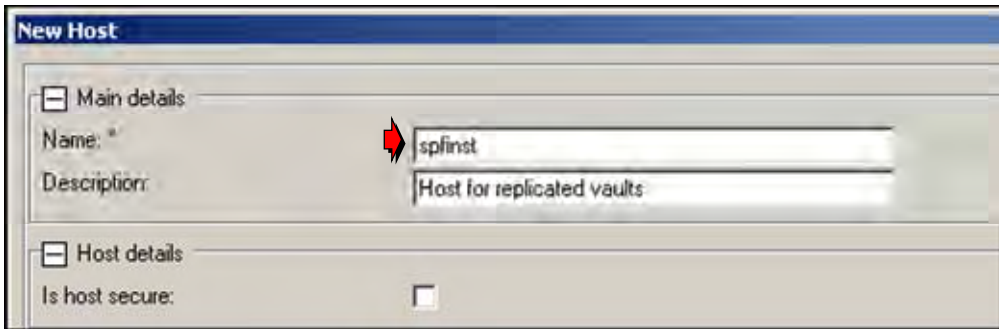
Use the *SmartPlant Foundation Desktop Client* to create the "local" **Host** and **Vault** object instances.

First, create the Host ohject that will be used as the local (remote) file server.

INTERGRAPH

Vault Replication

- **From the Desktop Client on the main server, select *File* > *New* > *Administration* > *Host* from the dynamic menu.**

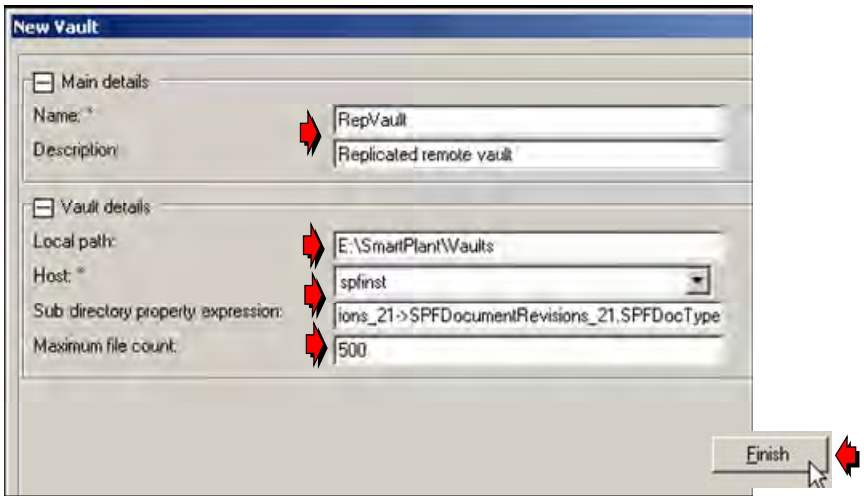


© 2008. Intergraph Corp.
All Rights Reserved.

Next, create a new vault object for the newly created local host.

Vault Replication

- **From the Desktop Client on the main server, select *File > New > Administration > Vault* from the dynamic menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

Click **Finish** to create the new replicated “local” vault.

For vault replication, you will need to configure additional relationships.

Users are associated with an organization or company. Additionally, a host or a collection of hosts are associated with an organization or company. Users have relationships to hosts through their relationship to a company or organization. This set of relationships provides users access to printer group and printers (discussed later) and replicated vaults.

Vault Replication

When using vault replication:

- **You must specify which file server is used to view the file.**
- **A relationship definition exists to link hosts and companies.**
- **A collection of file hosts are associated with a company object, which is then associated with one or more users. Users associated with that company have access to that collection of file hosts.**
- **If the host of the file matches or is in the set of hosts that are related to the user's company, it can be viewed from its original location as this host has been defined to be local to the current user.**

© 2008. Intergraph Corp.
All Rights Reserved.

Vault Replication

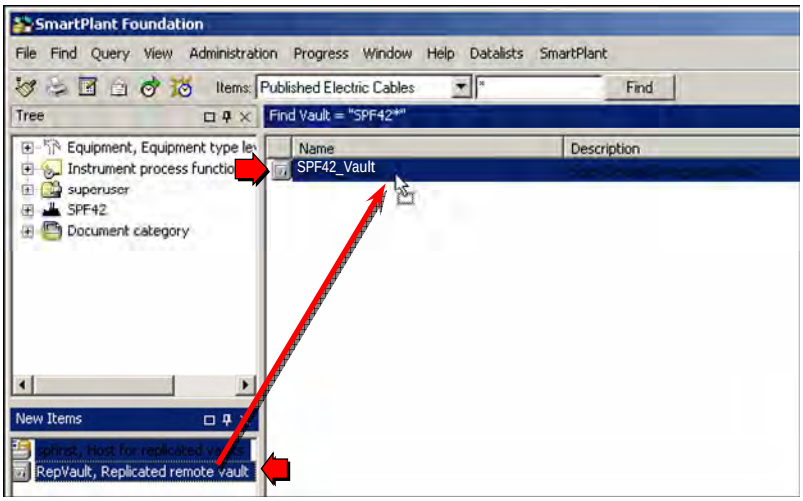
- If the host of the file is not in the set of hosts that are related to the user's company, the software will check to see if the vault is replicated to a host that *is* local to the current user. If so, the file can be viewed from this local host. If not, the file must be viewed remotely from the vault from which the file originates.
- If a User has no relationships with Company objects, or has relationships with Companies that have no relationships with hosts, the file will have to be viewed from its original location.

© 2008. Intergraph Corp.
All Rights Reserved.

Perform a search to locate the main vault then use a drag and drop to create a **SPFVaultIsReplicatedTo** relationship (RelDef).

Vault Replication

- Associate the new replicated vault with the main vault using the drag and drop operation.

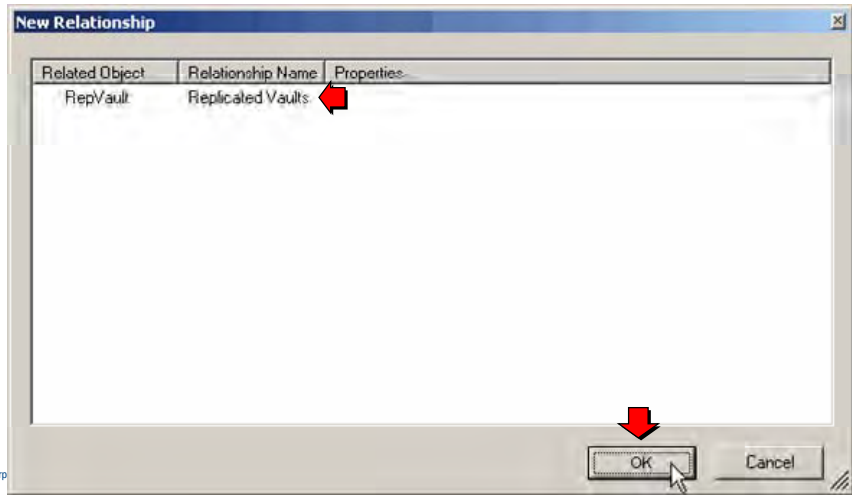


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Relationship* form will appear.

Vault Replication

- **On the confirmation screen, click *OK* to create the relationship.**

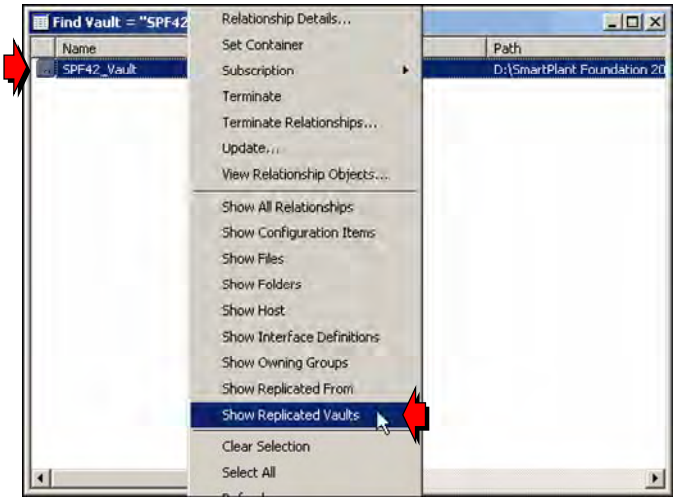


© 2008. Intergraph Corp.
All Rights Reserved.

Verify that the main vault can be replicated to the local (remote) vault via the new relationship.

Vault Replication

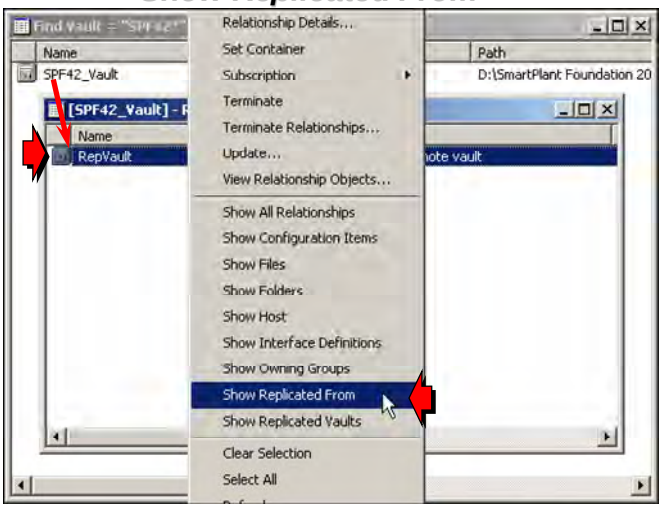
- **Right-click on the main vault and choose *Show Replicated Vaults* from the dynamic menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

Vault Replication

- **From the dynamic menu, right-click on the replicated vault and choose *Show Replicated From*.**

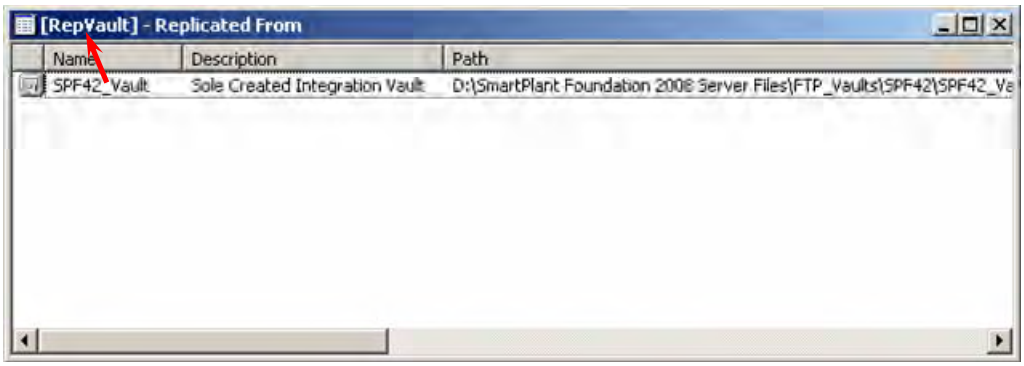


© 2008. Intergraph Corp.
All Rights Reserved.

A *Replicated From* form will appear to verify the **SPFVaultIsReplicatedTo** RelDef.

Vault Replication

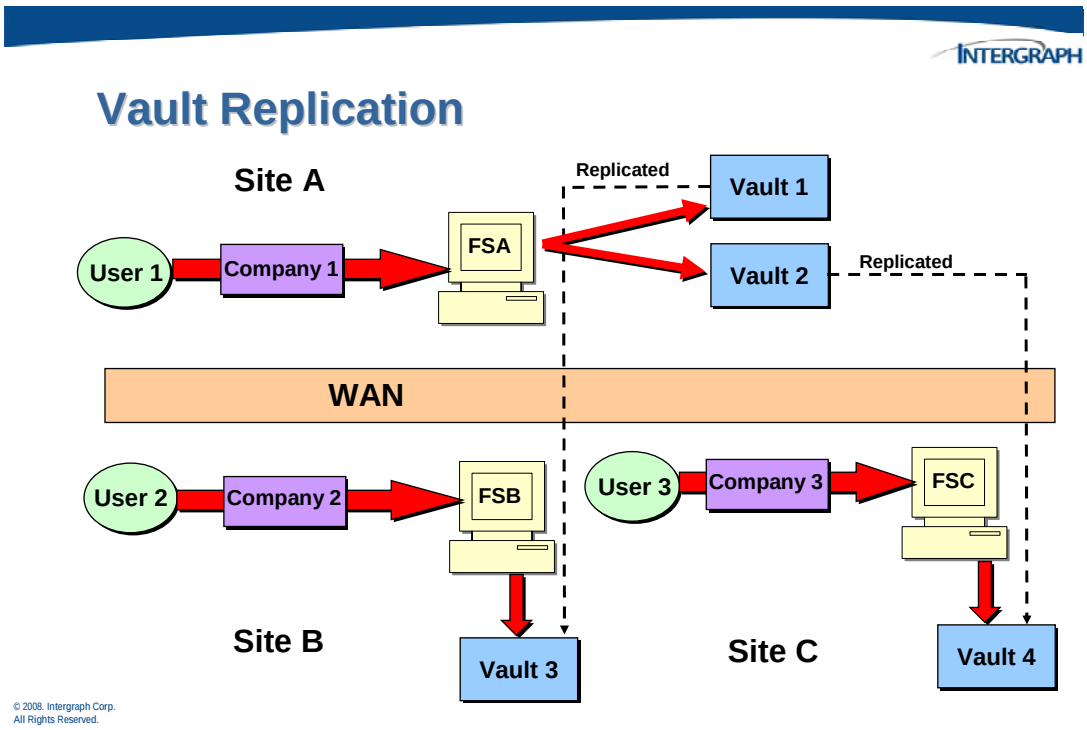
- **The *Replicated From* form will display to show the SPFVaultIsReplicatedTo RelDef.**



© 2008. Intergraph Corp.
All Rights Reserved.

3.12.3 Replicated Vault Viewing

SmartPlant Foundation replicated vaults only apply to the process of **viewing**. All other file operations occur directly back to the "Main Fileserver". The following flowchart shows the "*View and Markup*" options when replication has been enabled.



When a user selects a document for viewing, the system will find th file in the original vault (main vault). If that vault has a **SPFVaultIsReplicatedTo** relationship, then a check is made by the software to determine if the file to be viewed is also available in the replicated vault. If not, then the view will be viewed from the main vault.

In the above example:

- Site A is the main site where the one file server (FSA) hosts two vaults (Vault 1 and Vault 2).
- The file server (a host object) has a relationship with the company (Company 1).
- User 1 has a relationship with Company 1.
- When User 1 attempts to view a file that resides in Vault 1, the code checks to see if the file's vault (Vault 1) is on a server that is exposed by the user's company (Company 1). In this case, the condition is satisfied, so the user views the file directly from local Vault 1.

- Vault 1 is also replicated to a vault named Vault 3 on file server (FSB) at Site B. FSB is related to Company 2. User 2 belongs to this company.
- When User 2 attempts to view a file from Vault 1 (at Site A), the code will check to see if Vault 1 is on a server that is exposed by the user's company. Since FSA is not related to User 2's company (Company 2), the condition is not satisfied.
- Therefore, the code checks if Vault 1 (where the file originates) is replicated to a host which User 2 can access through his company (Company 2).
- Since Vault 3 is a replication of Vault 1 and is on a file server that User 2 can access through his company, User 2 can now view the file from User 2's local file server (FSB).
- When User 3 attempts to view a file that also originates in Vault 1, Vault 1's file server is also not found to be related to User 3's company. The same checks are made, and since Vault 1 is not replicated to a file server that User 3's company is related to, then no local vault has been found. In this case, User 3 has to view the file from its original vault, Vault 1.

3.12.4 File Replication Mechanisms

Replication is achieved by using a third party application. For testing purposes, **Robocopy** was used as the replication mechanism, however, any 3rd party replication software can be used.

Note: The replication must replicate the files to the same physical location as the source vault as all the replicated vaults use the same vault information and hence need to be located in the same location

The following example demonstrates full replication where **All** data is copied from the source to the destination with two types of synchronization:

- Constant Synchronization - data is constantly synchronized.
- Periodic Synchronization - Data is periodically synchronized to reduce network bandwidth.

3.12.5 Using Robocopy for SPF Replication

Below is an example of a file replication package (there are numerous packages on the market that can be used).

Robocopy (Robust File Copy Utility) is a 32bit command-line tool used for file replication and is included in *Microsoft® Windows® Resource Kit*.

The version used here is version XP010 which requires Microsoft® Windows® Server 2003, Microsoft® Windows® XP, Microsoft® Windows® 2000, or Microsoft® Windows NT® version 4.0.

To run Robocopy, use the following syntax at the command prompt:

ROBOCOPY source destination [file [file]...] [options]

The following table defines these syntax elements.

Variable	Description	Comments
source	Specifies the path to the source directory.	You can use *drive*:*path* or *\\server\share\path*
destination	Specifies the path to the destination directory.	You can use *drive*:*path* or *\\server\share\path*
File	Specifies the file or files to be copied.	You can use wildcard characters (* or ?), if you want. **If the File parameter is not specified, *.* is used as the default value.**
options	Specifies options to be used with the robocopy command.	Available options are described later in this document.

Copy options:

Parameter	Description
Source	Source directory (can be UNC)
destination	Destination directory (can be UNC
File	Names of files to act upon
Options	--o—
/S	Copies Subdirectories (excluding empty ones)
/MOT:M	Monitors source, and runs again in *M* minutes if changes are detected
/MON:N	Monitors the source, and runs again when more than *N* changes are detected

/MIR	Mirrors a directory tree (will delete at destination)
/NDL	Specifies that directory names are not to be logged
/LOG+:<logfile>	Writes the status output to the log file (appends the output to the existing log file)
/TEE	Writes the status output to the console window, as well as to the log file

i.e.

robocopy \\spfserv\vaults \\fileserv\vaults c:\apd\dir1 **.** /S /MOT:1 /MIR /NDL /TEE

This would replicate every minute as long as a file has changed.

3.12.6 Constant Synchronization

This is achieved by setting one of the monitoring options to 1

i.e. /MON:1 or /MOT:1

The command:

robocopy \\spfserv\vaults \\fileserv\vaults c:\apd\dir1 . /S /MOT:1 /MIR /NDL /TEE

This would replicate every minute if a file change is detected.

3.12.7 Periodic Synchronization

Robocopy copy can be controlled to run in several different ways.

The following option can be used to run at a **Constant interval:**

i.e. **/MOT:30**

- Will make robocopy run every 30 minues

Restricted hours:

i.e. **/RH:2400:0400**

- Will restrict robocopy to only run between the hours of 24:00 and 04:00

You can also control the following:

Bandwidth Restriction:

i.e. **/IPG:100**

- Will put a 100 millisecond delay between each robocopy packet (normally 64 kilobyte)

File Size:

i.e. **/MAX:<1000000>**

- Will exclude files of size larger than1000000 bytes (1MB).

Batch jobs - the robocopy can be run as a batch job (AT or “Scheduled Tasks”)

3.12.8 Recovery options

In case on failure of a remote file location, robocopy can be made to wait and retry before failing . (The default is a million retries of 30 seconds about a year).

/R:n specifies number of retries

/W:secs specifies the wait in seconds

3.12.9 Installation as a Service

Robocopy can be installed as a service using another *Microsoft® Windows® Resource Kit* utility - “svrany”.

3.13 Activity 2 – SPF Scheduler and Calendars

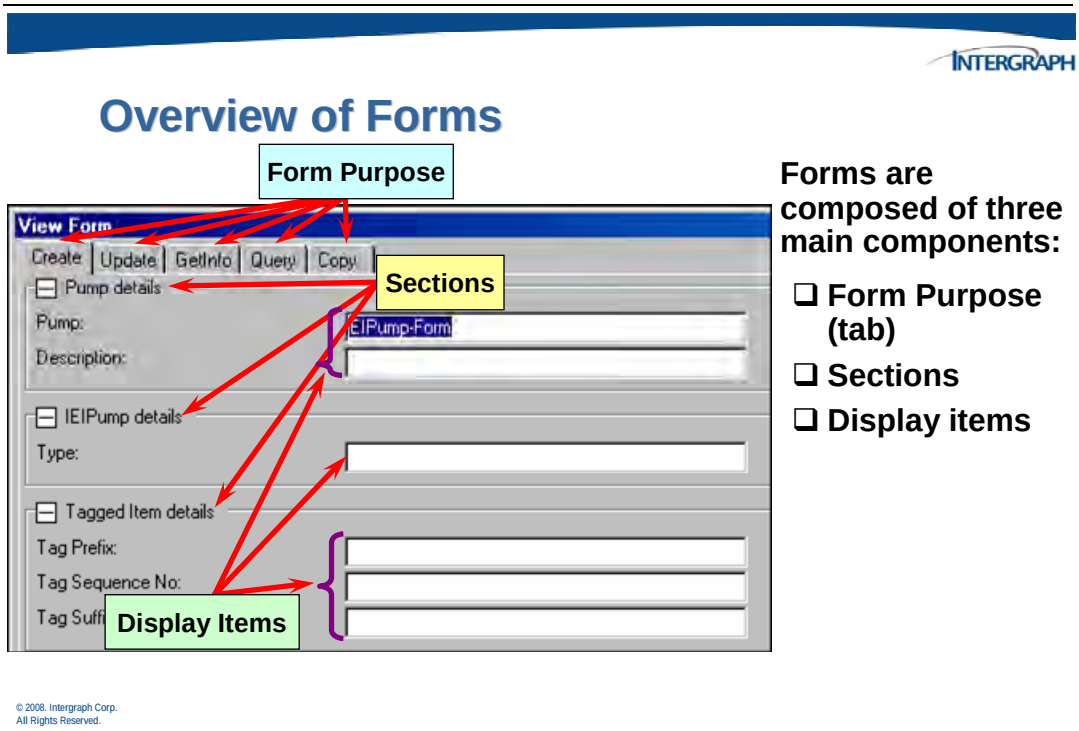
Complete **Chapter 3 - Activity 2** in the SPF Configuration and Administration II activity workbook.

CHAPTER 4

Creating SmartPlant Foundation Forms

4. Overview of Forms

SmartPlant Foundation uses forms to provide support for creation, update, query, clone, and information display of all objects such as Tag, Asset, Models, and Documents.



The form purpose controls the form sections structure based on its purpose. The purpose is displayed as a tab on the view form.

Sections define a collection of display items (properties or relations). Sections may be either created or re-used from the database. Once a section is created, display items can be added to the section.

A section can be attached to multiple forms with any changes to that section automatically updated on all forms.

Display items are used to present data or relationships on a form. After a display item is created, the display item can then be added to a section for inclusion on a form. A display item can be a member of multiple sections. If a display item is updated, the changes are automatically updated on the forms.

Overview of Forms

The best way to visualize the form that gets converted from the schema definitions that were loaded into the database is as follows:

- **ClassDef = Form**
- **Interfaces = Sections**
- **Properties = Display Items**

© 2008. Intergraph Corp.
All Rights Reserved.

4.1 Generating a Form

There are several ways to create a form.

Forms can be created by:

- ❑ Generating a form for a class definition.
- ❑ Using the New Form form to manually create the form.
- ❑ Copying and editing an existing form

You can generate forms for class definitions contained in the SmartPlant schema. These forms contain all the property definitions that are exposed by the interface definitions realized by a particular class definition.

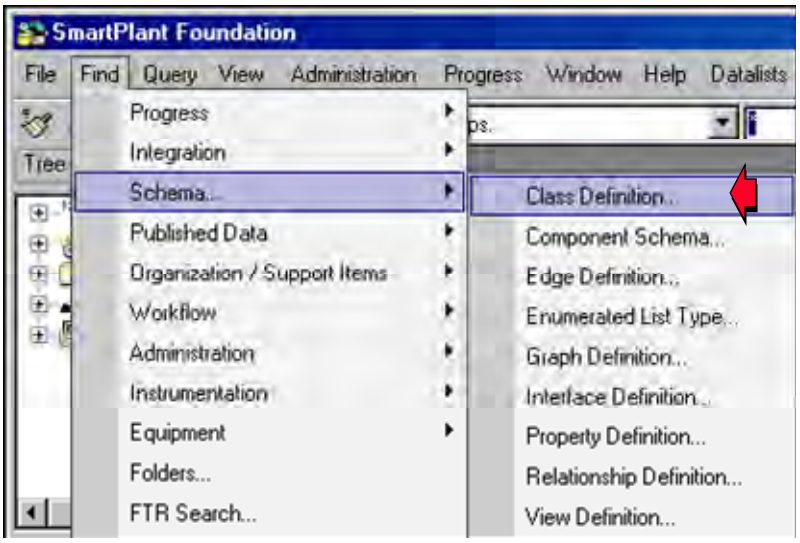
Once a form has been generated, sections and display items can be copied and modified to customize the form appearance.

All of the different ways of creating forms will be discussed in this chapter.

INTERGRAPH

Generating a Form from a Class Definition

❑ **From the menu, click *Find > Schema > Class Definition*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Class Definition* dialog will appear.

Generating a From from a Class Definition

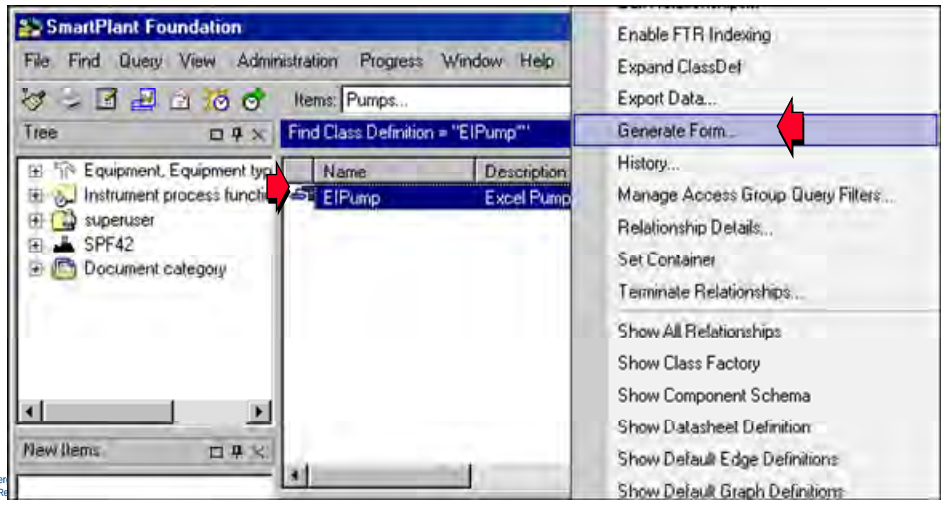
- **Enter *ElPump* and the wildcard character, *, to find and display all *Class Definitions* that start with ElPump, then click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

A list view window will display containing the results of the find.

Generate Form

- **Right click on the *ElPump* class definition and select *Generate Form* from the shortcut menu.**



© 2008. Inter
All Rights Re

Forms can be generated via shortcut menu commands. The following rules apply:

- A common *Name Description* and *General Information* section is used on all objects.
- Administration forms include the *ContainerID* in the Name Description section.
- Additional display items are added for configuration controlled classes to display the query and existing configuration of the object.
- Sections and Display items are generated as follows:
  - A **Section** is generated for any Interface that:
    - Has a realizes order value > -1 (so the model can control forms).
    - Contains properties or relationships that need display items (see below).
  - A **Display item** is generated for:
    - Properties with an Exposes order value > -1.
    - Required relationships: Check cardinality (Include End1 RelDefs if the min2 = 1 or max2 = 1. Include End2 RelDefs if min1 = 1).
- Do not include abstract relationships.
- If the relationships have a link interface, then always include any Relationship Definitions on the link interface.
- An additional section is generated with display items for optional interfaces that:
  - Have a realizes order value of > -1 and have no properties or relationships.

4.2 Form Editing with Form Builder

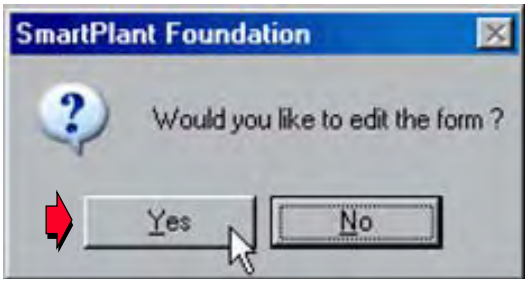
The SPF configuration can vary widely from customer to customer. The type and format of data also varies and must be displayed dynamically. To accomplish this, **forms** can be created and modified using the *Form Builder*.

- ❑ Create, Update, Query, GetInfo (viewing), Copy and Revise forms functionality is available

When generating a form from a class definition, click **Yes** when prompted to edit the form.

Form Builder

- ❑ **To edit the form with *Form Builder*, click Yes.**



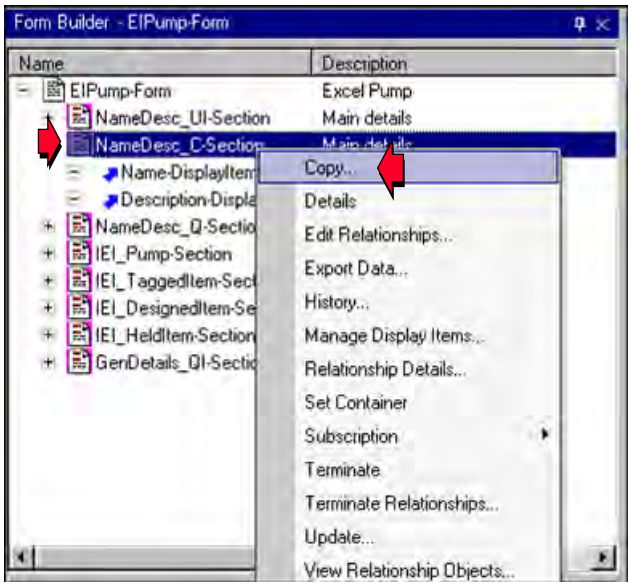
© 2008. Intergraph Corp.
All Rights Reserved.

The **Form Builder** is a convenient way to display the form structure and access the *Manage Sections* and *Manage Display Items* forms.

The Form Builder is displayed in the *List View* pane.

Form Builder

- **Right click on the *NameDesc_C* section and select *Copy* from the shortcut menu.**

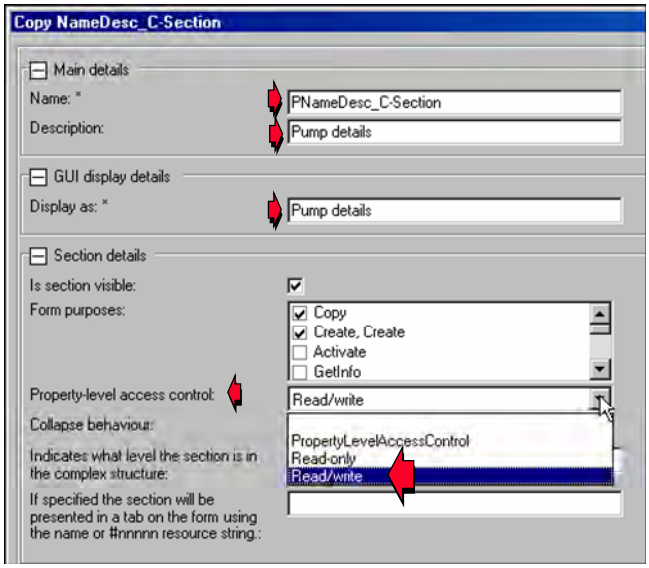


© 2008. Intergraph Corp.
All Rights Reserved.

Copying a section allows you to create a new section definition that references the same display items as the original section. It also copies all the associated relationships.

Form Builder

- **Change the appropriate fields to define a new *Form Section* based on the selected section.**
- **Select Read/Write for *Property-level access control*.**



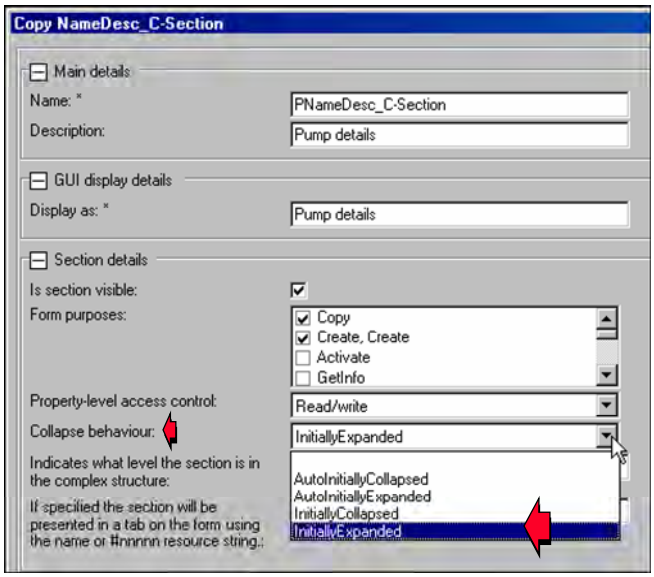
© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Section* form contains the following fields:

- **Name** - Specifies the name/purpose of the section and is a required field.
- **Description** - Specifies a description for the section and is an optional field.
- **Display as** - Specifies the section name as it should appear on the form. This is a required field.
- **Is section visible** - Select to make the section visible in the form.
- **Form purposes options**-
  - **Activate** - Section is used when activating a reserved document (i.e. Vendor documents). This is different from create because some of the properties set during document reservation might need to be read only during activate (e.g. name).
  - **Copy** - Section is to be displayed only on copy forms. When an object is copied, only those display items that are associated with a section that has the Copy option selected are copied.
  - **Create** - Section is to be displayed only on create and copy forms.
  - **Get info** - Section is to be displayed only on property forms or detail options. All display items are presented as read-only so the same section can be used for Update and Get Information.
  - **HisRev** - Section is to be displayed only on history revision forms.
  - **Query** - Section is to be displayed only on query forms.
  - **Reclassify** - Section is to be displayed only on reclassify forms.
  - **Revise** - Section is to be displayed only on revise forms.
  - **Sign Off** - Section is to be displayed only on sign off forms.
  - **Update** - Section is to be displayed only on update forms.
- **Property-level access control** - Overrides the display items setting. Options are:
  - **PropertyLevelAccessControl** - All display items in the section have individual access control.
  - **Read only** - All display items in the section are read-only regardless of their individual settings.
  - **Read/Write** - All display items in the section are editable regardless of their individual settings.

Form Builder

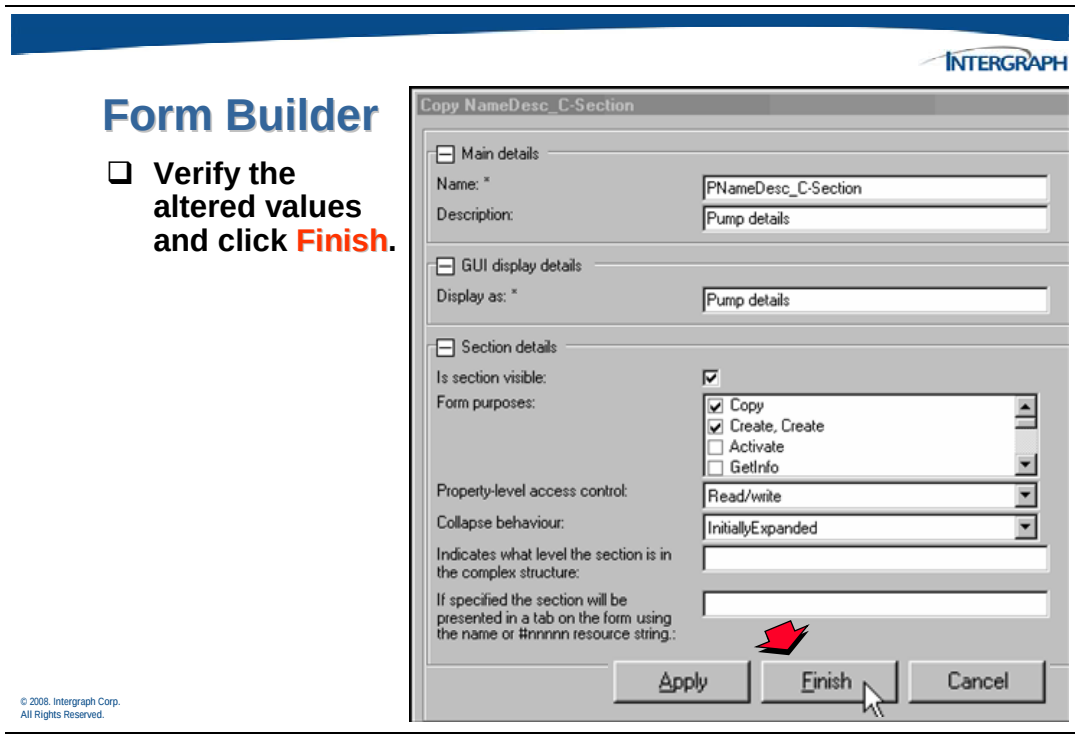
- **Select InitiallyExpanded for *Collapse behaviour*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Section* form fields con't:

- **Collapse behaviour** - Determines whether the form displays in collapsed or expanded mode. The options are:
  - Auto Initially Collapsed
  - Auto Initially Expanded
  - Initially Collapsed
  - Initially Expanded



The *Copy Section* form fields con't:

- **Indicate what level the section is in the complex structure** - Determines which object of a complex object to set the properties on. This is an optional field.
- **If specified section will be presented in a tab on the form using the name or #nnnn string** - String in which you can enter a name for the tab.

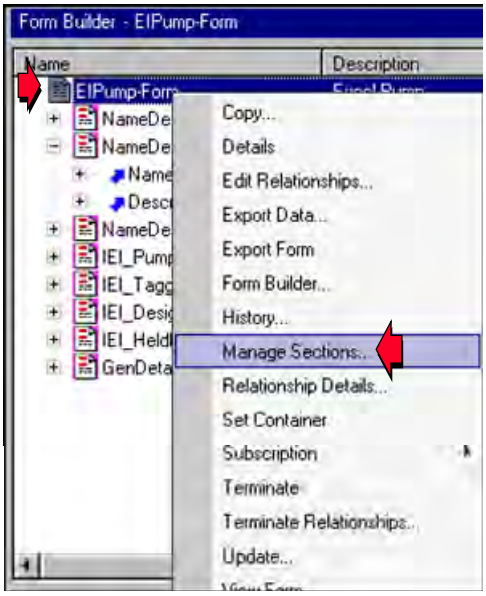
4.2.1 Modifying Data Form Sections

A form's sections can be managed through the *Manage Sections* Form. This form can be used to *create*, *update* and *re-order* sections, *terminate* relationships, and *delete* sections.

INTERGRAPH

Manage Form Sections

- **Right click on the *EIPump-Form* object and from the shortcut menu, select *Manage Sections*.**

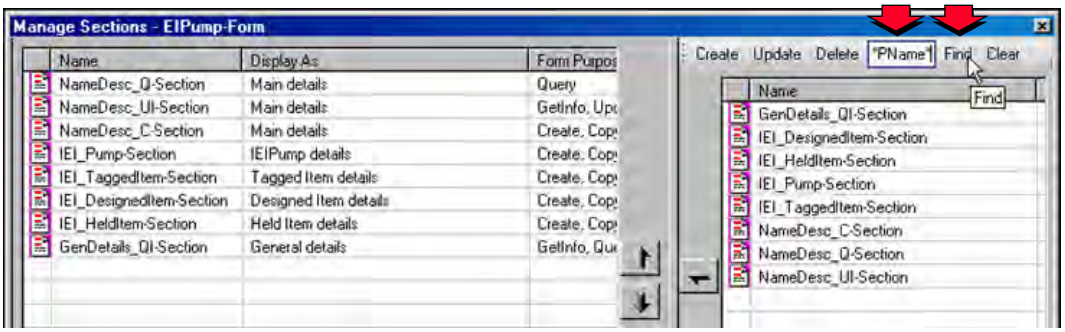


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Sections* form will appear.

Manage Form Sections

- **To find the new section, enter **PName** in the search field and click Find.**



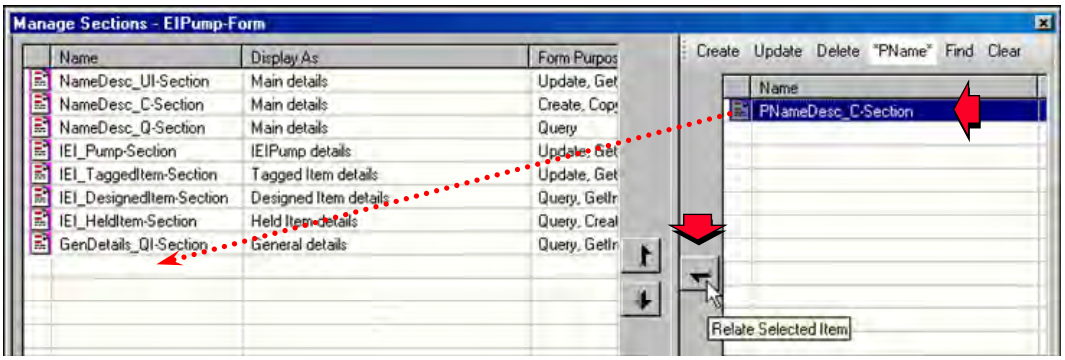
© 2008. Intergraph Corp.
All Rights Reserved.

The results are displayed in the **Library** pane (on the right).

Add the new section to the *EIPump* form.

Manage Form Sections

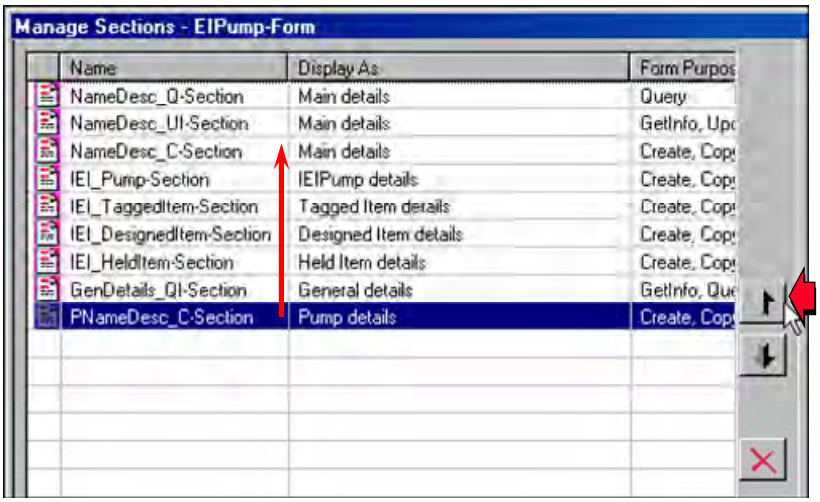
- **Select the *PNameDesc_C-Section* and click the left arrow button to relate the item to the EIPump-Form.**



© 2008. Intergraph Corp.
All Rights Reserved.

Manage Form Sections

- **Use the up arrow to move the new section above the original *NameDesc_C-Section*.**

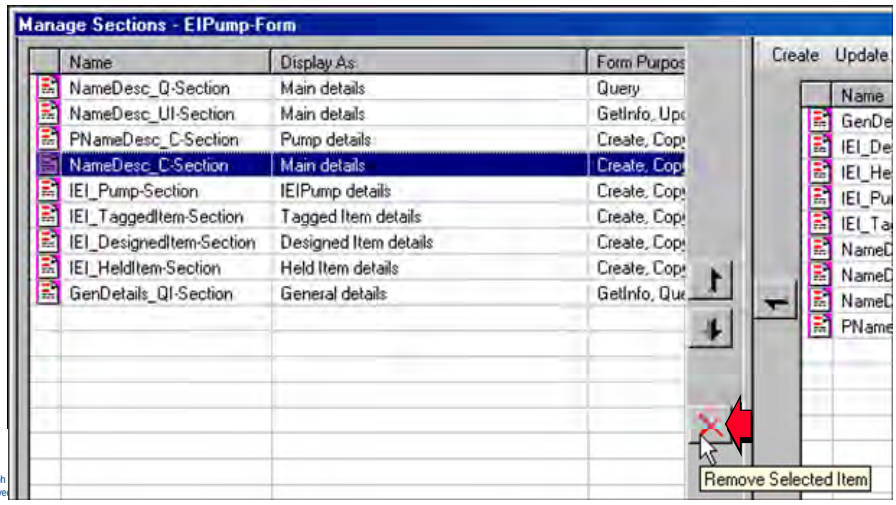


© 2008. Intergraph Corp.
All Rights Reserved.

Select the new section and use the arrows to change the section order.

Manage Form Sections

- **Select the original *NameDesc_C-Section* and click the delete button to remove the section from the form.**

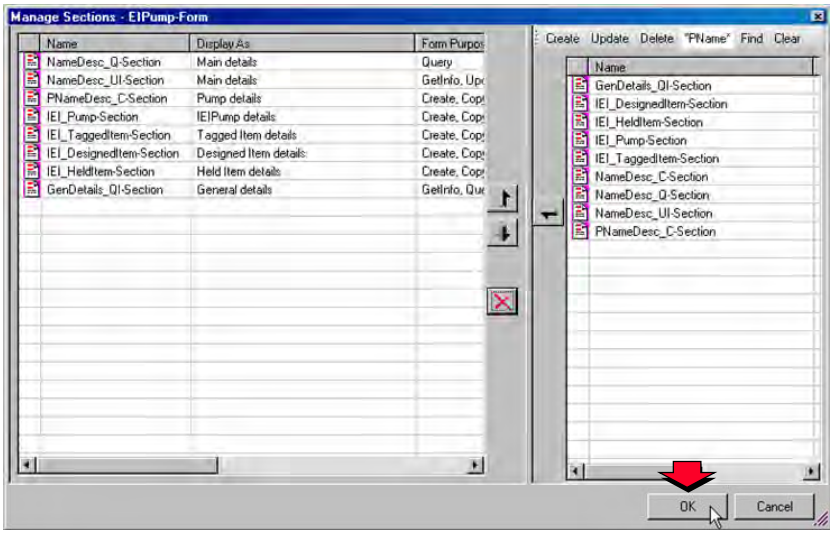


© 2008. Intergraph
All Rights Reserve

To delete a section from a form, it is only necessary to delete the relationship. The section will remain in the library for use by other forms.

Manage Form Sections

- **Click OK to save the changes to the form.**



© 2008. Intergraph Corp.
All Rights Reserved.

Next, an example of customizing the *Name* display item will be shown.

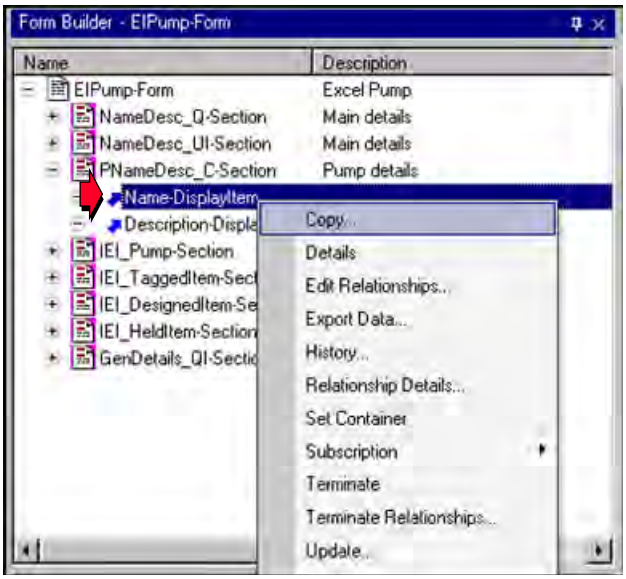
4.2.2 Modifying Display Items

Display items are used to present data or relationships on a data form. Properties or relations are assigned for a display item from a relational table. New display items can be created or display items can be re-used from the database. After a display item is created, the display item can then be added to a section for inclusion on a data form.

Make a copy of the existing **Name** display item so that any changes will only affect the *EIPump* form.

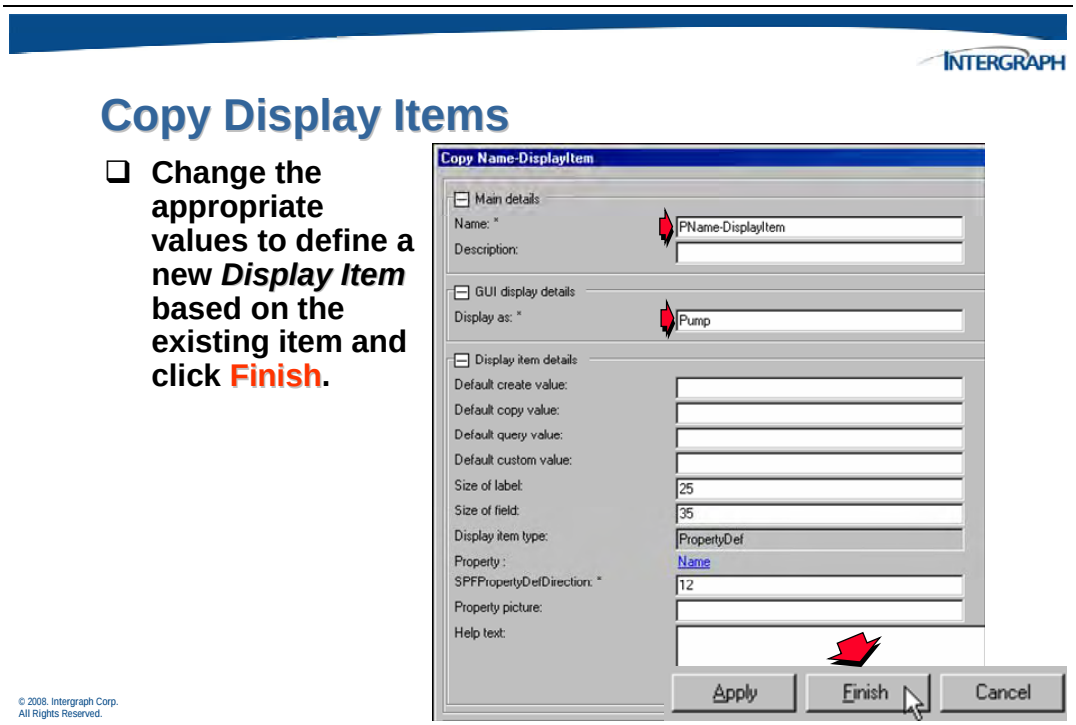
Copy Display Items

- **Right click on *Name-DisplayItem* and from the shortcut menu, select *Copy*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Name DisplayItem* form will appear.



The *Copy Section* form contains the following fields:

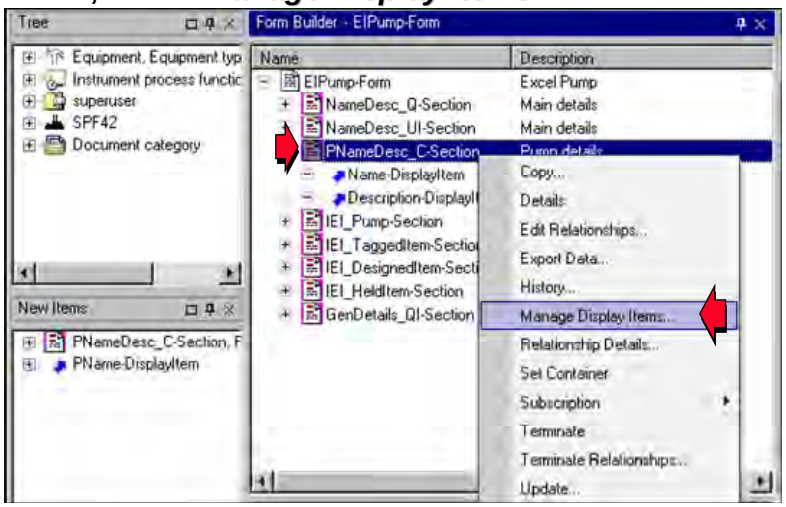
- **Name** - Display item name and is a required field.
- **Description** - Display item description.
- **Display as** - The name the item will display on the form and is a required field.
- **Default create value** - The default value for the display item in *New* dialog boxes that contain this display item.
- **Default Copy Value** - The default value for the display item in *Copy* dialog boxes that contain this display item.
- **Default Query Value** - The default value for the display item in *Query* dialog boxes that contain this display item.
- **Default Custom Value** - The default value to use on special forms.
- **Size Of Label** - The length of the display name.
- **Size Of Field** - The input field size. (The input is truncated to the specified size before it is stored.)
- **Display item display type** - Select an object filter to limit the number of related objects listed.
- **Property Picture** - A format for the display item data. The format can either be U for uppercase or any picture mask as defined in Picture Mask Types.

- **Help text** - Help text for display item. (The text to appear when the mouse is hovered over the display item.)

The display items in a section can be managed through the *Manage Display Items* form. This form can be used to *create*, *update* and *re-order* display items, *terminate* relationships, and *delete* display items.

Manage Form Sections

- **Right click *PNameDesc_C-Section* and from the shortcut menu, select *Manage Display Items*.**

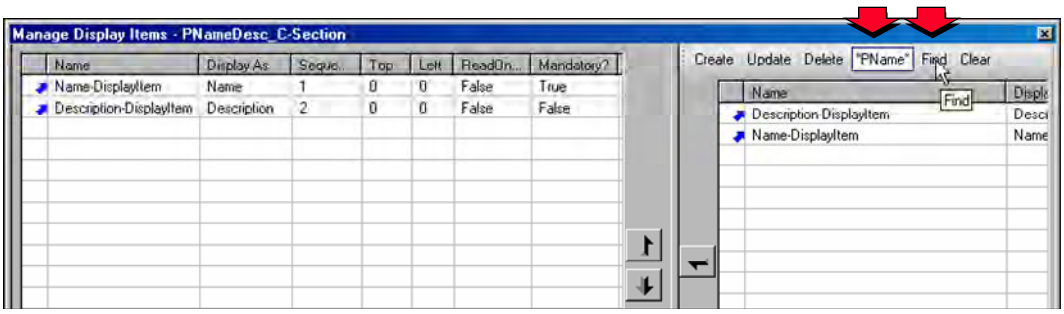


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Display Items* form will appear.

Manage Display Items

- **To find the new display item, enter **PName** in the search field and click Find.**

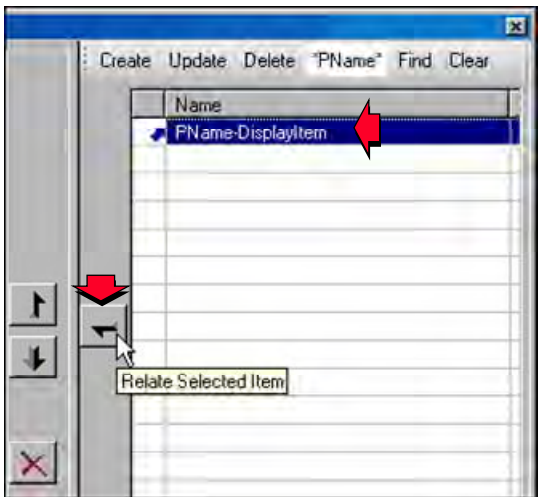


© 2008. Intergraph Corp.
All Rights Reserved.

The results are displayed in the *Library* pane.

Manage Form Sections

- **Select *PName-DisplayItem* and click the left arrow button to relate the item to the *PNameDesc_C-Section* section.**

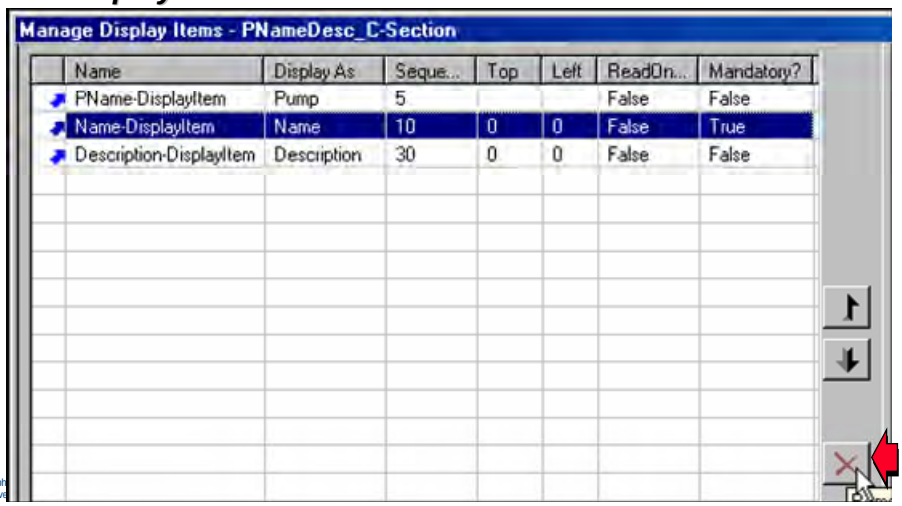


© 2008. Intergraph Corp.
All Rights Reserved.

Change the *DisplayAs* value for the display item to reflect the name as **Pump** as shown below.

Manage Form Sections

- **Use the delete button, to delete the original *Name-DisplayItem*.**

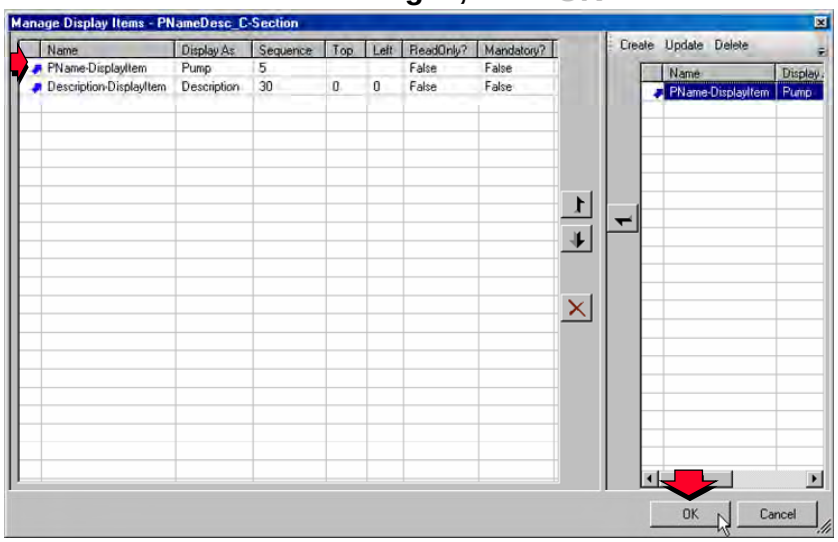


© 2008. Intergraph
All Rights Reserve

To delete a display item from a section, it is only necessary to delete the relationship. The display item will remain in the library for use by other sections.

Manage Form Sections

- **To save the section changes, click OK.**

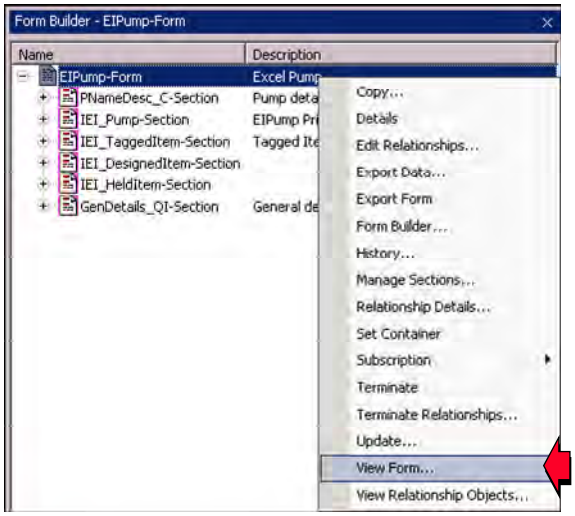


© 2008. Intergraph Corp.
All Rights Reserved.

Finally, save the changes in the *Manage Display Items* form.

Manage Form Sections

- **Right click on *EIPump-Form* and from the shortcut menu, select *View Form*.**



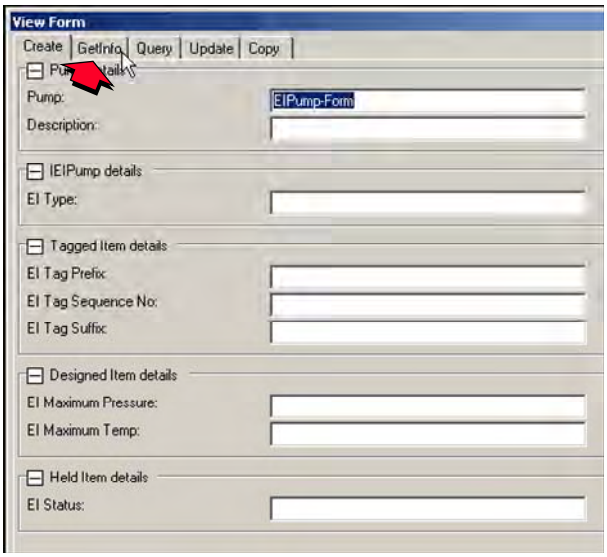
© 2008. Intergraph Corp.
All Rights Reserved.

This will display the form with the new custom section and display item.

The **Main details** section is actually the *PNameDesc_C-Section* copied section. By making a copy of this section, the display labels can be changed without affecting other forms that use the default *NameDesc_C-Section*.

Manage Form Sections

- **From the *View Form* click the GetInfo tab.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *GetInfo* form will be displayed.

Manage Form Sections

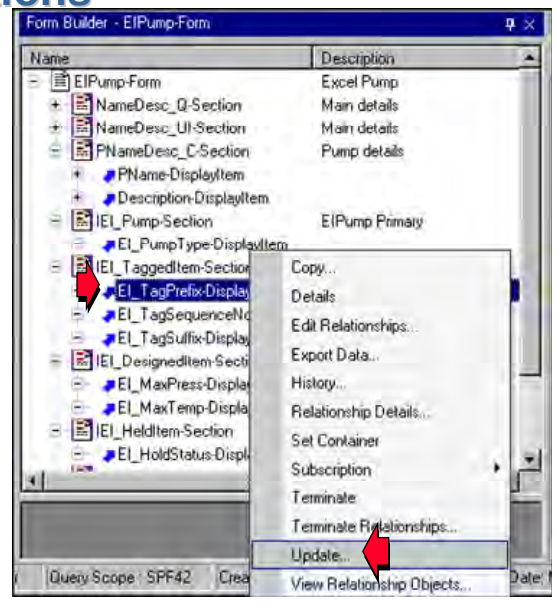
- **The available display items for all sections of the form are displayed under the *GetInfo* tab.**
- **Click Close to dismiss the form.**

© 2008. Intergraph Corp.
All Rights Reserved.

To change the display labels that display on the forms, use the ***Update*** object command.

Manage Form Sections

- **Right click on *EI_TagPrefix-DisplayItem* and select *Update* from the shortcut menu.**

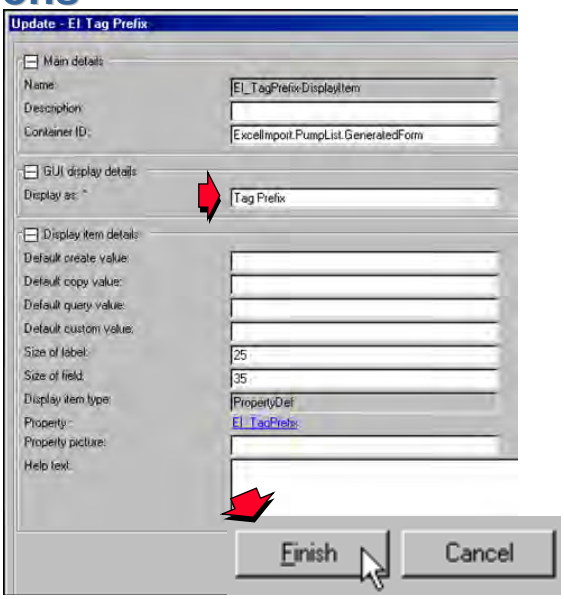


© 2008. Intergraph Corp.
All Rights Reserved.

The *Update* form will appear.

Manage Form Sections

- **Change the *Display as* field to the value *Tag Prefix* and click Finish.**



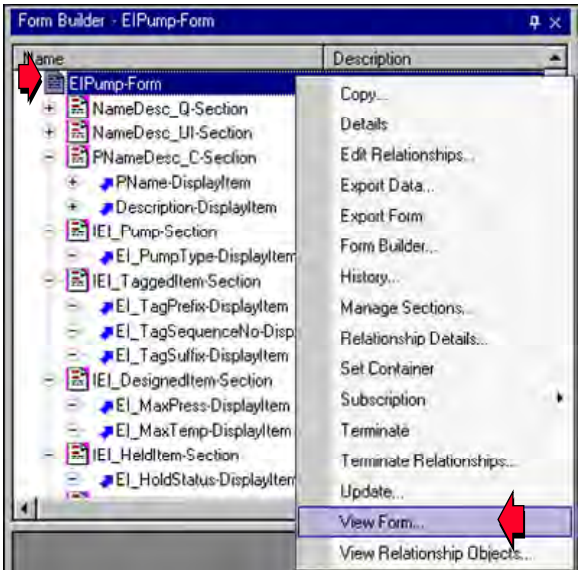
© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the **Update** process for the other labels *Type*, *Tag Prefix*, *Tag Sequence No.*, *Tag Suffix*, *Maximum Pressure*, *Maximum Temp*, and *Status*.

After the changes have been made, view the form to verify the modifications.

Manage Form Sections

- **Right click on the EIPump-Form object and select *View Form* from the shortcut menu.**

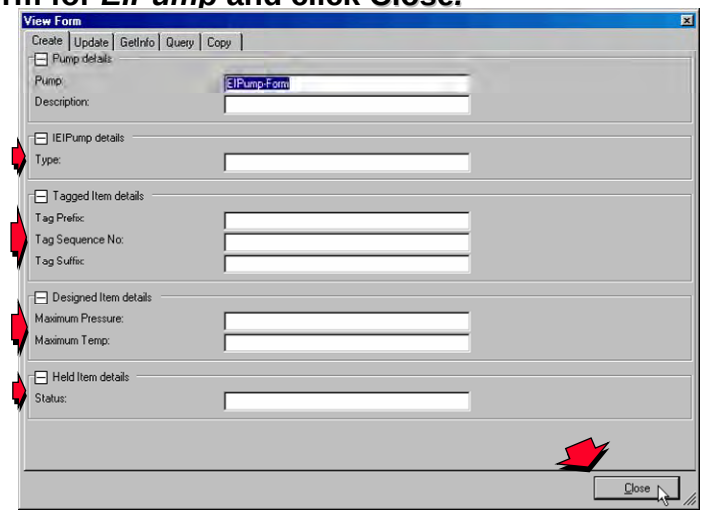


© 2008. Intergraph Corp.
All Rights Reserved.

The *View Form* form will appear.

Manage Form Sections

- **Verify that all of the Display as changes are reflected on the form for *EIPump* and click Close.**



© 2008. Intergraph Corp.
All Rights Reserved.

4.3 Activity 1 – Creating Forms

Complete **Chapter 4** - **Activity 1** in the SPF Configuration and Administration II activity workbook.

4.4 Creating Forms Manually

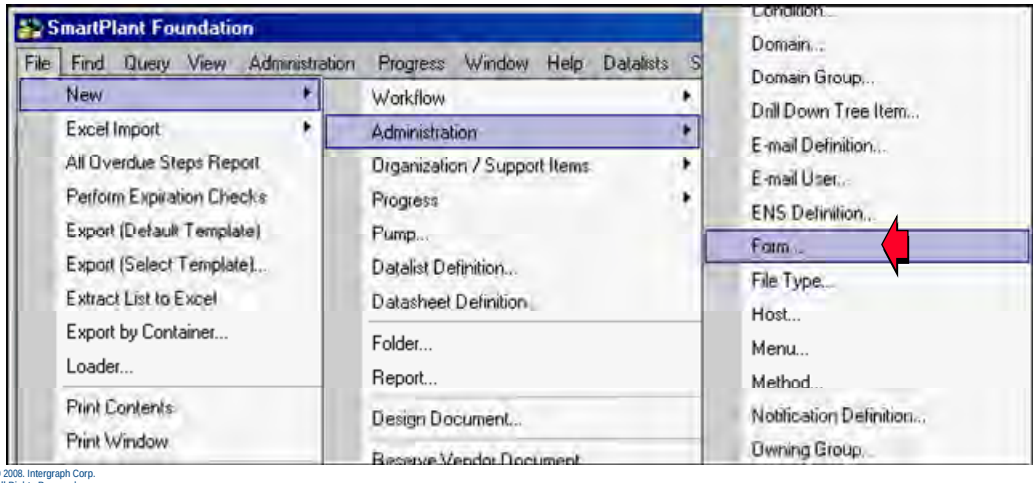
In addition to generating forms based on Class Definitions and Interface Definitions, forms can also be created manually. This will involve creating the *form* object, each of the *sections* that relate to the form and finally the *display items* that make up each section.

To create a new form manually:

INTERGRAPH

Creating a Form Manually

- **From the menu, click *File > New > Administration > Form*.**

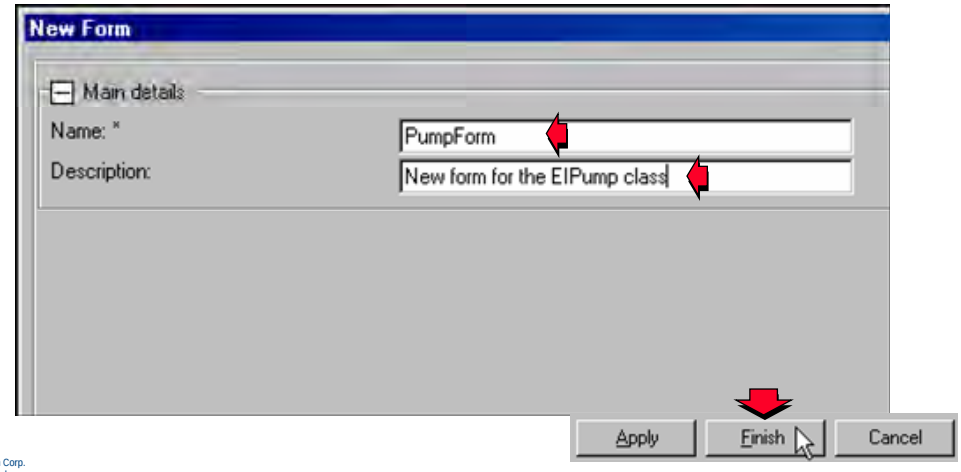


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Form* form will appear.

Creating a Form Manually

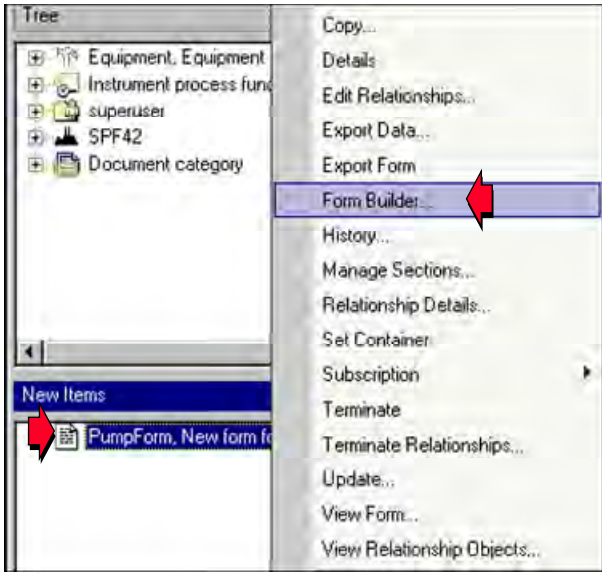
- **Enter a required form *Name* and an optional *Description* and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

Manually add the required *sections* and *display items* to the new form using **Form Builder**.

Creating a Form Manually



- **Right click on the new *PumpForm* and select *Form Builder* from the shortcut menu.**

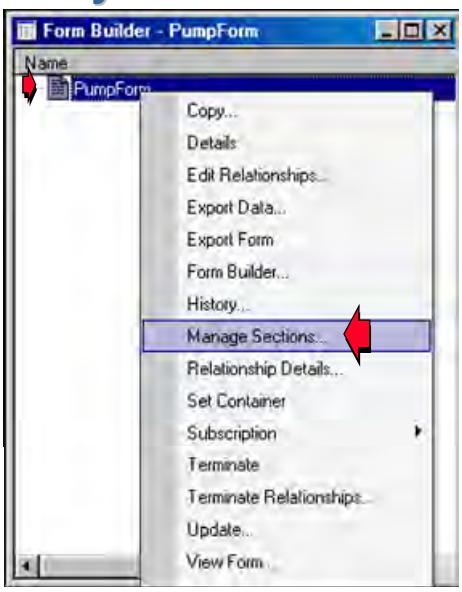
4.4.1 Creating Sections Manually

Sections define a collection of display items (properties or relations). Sections may be either created or re-used from the database. Once a section is created, display items can be added to the section.

A form's sections can be created through the *Manage Sections* Form.

Creating a Section Manually

- **Right click on the *PumpForm* and select *Manage Sections* from the shortcut menu.**

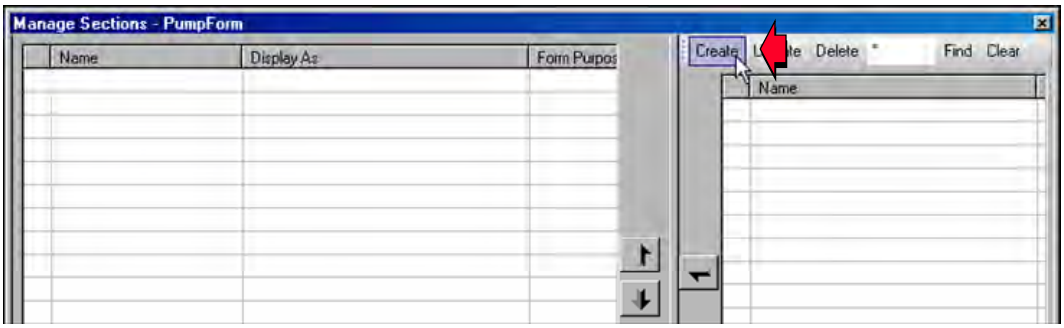


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Sections* form will appear.

Creating a Section Manually

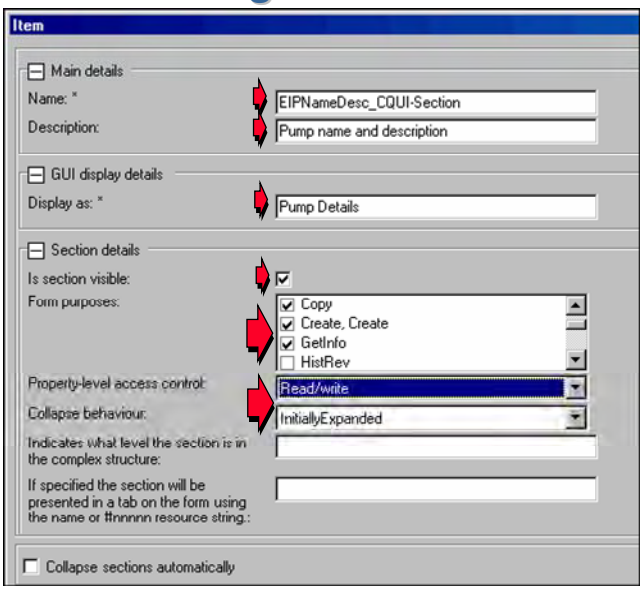
- **Select *Create* from the menu to define a new form section.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Item* form will appear.

Creating a Section Manually



- **Enter the information needed to create a new form section and click Finish.**

The *Item* form contains the following fields:

- **Name** - Specifies the name/purpose of the section and is a required field.
- **Description** - Specifies a description for the section and is an optional field.
- **Display as** - Specifies the section name as it should appear on the form. This is a required field.
- **Is section visible** - Select to make the section visible in the form.
- **Form purposes options**-
  - **Activate** - Section is used when activating a reserved document (i.e. Vendor documents). This is different from create because some of the properties set during document reservation might need to be read only during activate (e.g. name).
  - **Copy** - Section is to be displayed only on copy forms. When an object is copied, only those display items that are associated with a section that has the Copy option selected are copied.
  - **Create** - Section is to be displayed only on create and copy forms.
  - **Get info** - Section is to be displayed only on property forms or detail options. All display items are presented as read-only so the same section can be used for Update and Get Information.
  - **HisRev** - Section is to be displayed only on history revision forms.
  - **Query** - Section is to be displayed only on query forms.
  - **Reclassify** - Section is to be displayed only on reclassify forms.
  - **Revise** - Section is to be displayed only on revise forms.
  - **Sign Off** - Section is to be displayed only on sign off forms.
  - **Update** - Section is to be displayed only on update forms.
- **Property-level access control** - Overrides the display items setting. Options are:
  - **PropertyLevelAccessControl** - All display items in the section have individual access control.
  - **Read only** - All display items in the section are read-only regardless of their individual settings.
  - **Read/Write** - All display items in the section are editable regardless of their individual settings.
- **Collapse behaviour** - Determines whether the form displays in collapsed or expanded mode. The options are:
  - Auto Initially Collapsed
  - Auto Initially Expanded

- Initially Collapsed
- Initially Expanded

❑ **Indicate what level the section is in the complex structure** - Determines which object of a complex object to set the properties on. This is an optional field.

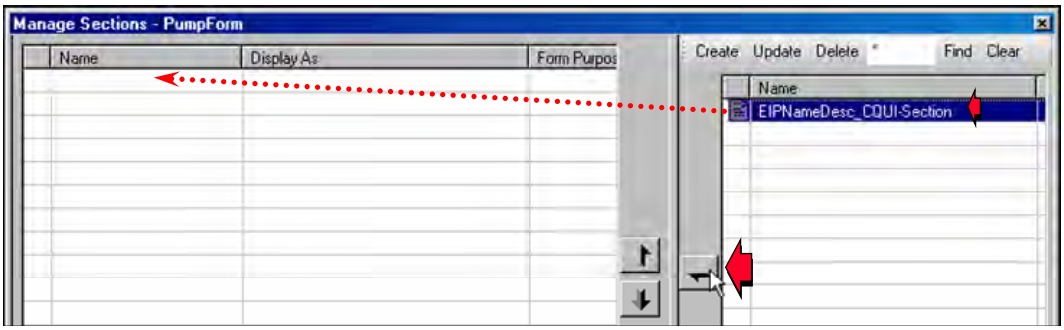
❑ **If specified section will be presented in a tab on the form using the name or #nnnn string** - String in which you can enter a name for the tab.

The new section is displayed in the *Library* pane.

INTERGRAPH

Creating a Section Manually

❑ **Select *EIPNameDesc_CQUI-Section* and click the left arrow button to relate the section to the form.**

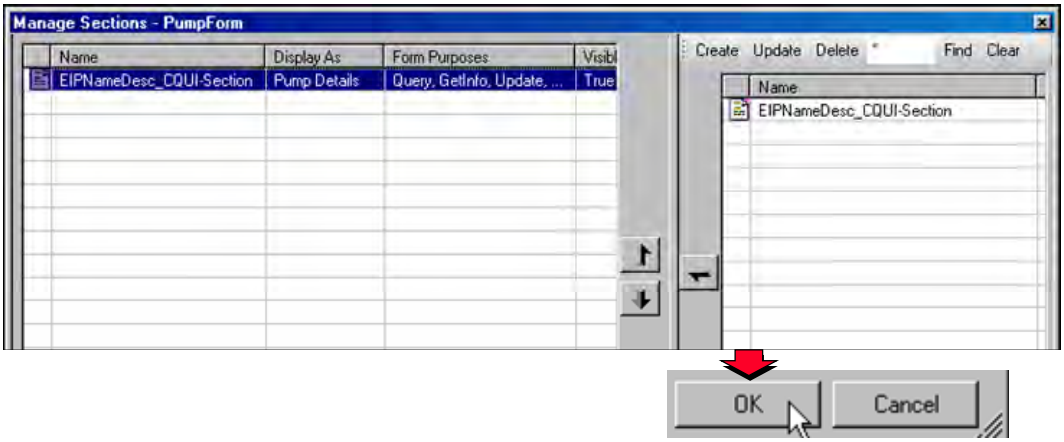


© 2008. Intergraph Corp.
All Rights Reserved.

Edit any of the relationship properties such as *DisplayAs*, *FormPurposes*, etc. to contain the desired information.

Creating a Section Manually

- **To save the new section relationship, click OK.**



© 2008. Intergraph Corp.
All Rights Reserved.

Once the section has been created, the display items can be added to it.

4.4.2 Creating Display Items Manually

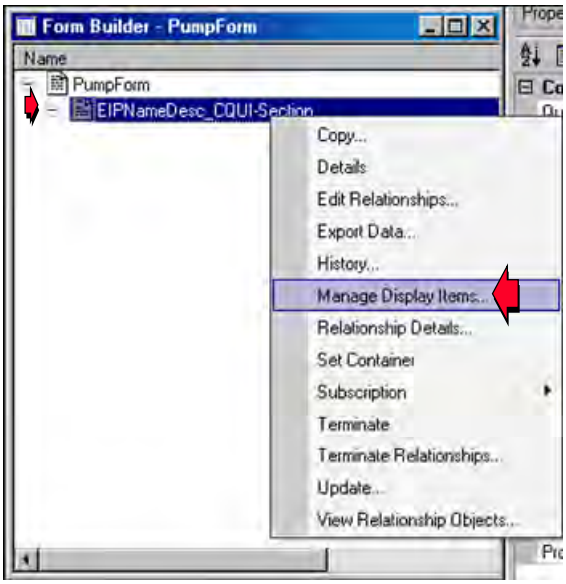
Display items are used to present data or relationships on a form. Properties or relations are assigned for a display item from a relational table. Users can create new display items or re-use display items from the database.

Once a display item is created, it appears in the right pane of the *Manage Related Items* form and it can then be added to a section for inclusion on a form. You can search for existing display items and sections to help you complete your form.

INTERGRAPH

Creating a Display Item Manually

- **Right click on the *EIPNameDesc_CQUI-Section* and from the shortcut menu, select *Manage Display Items*.**

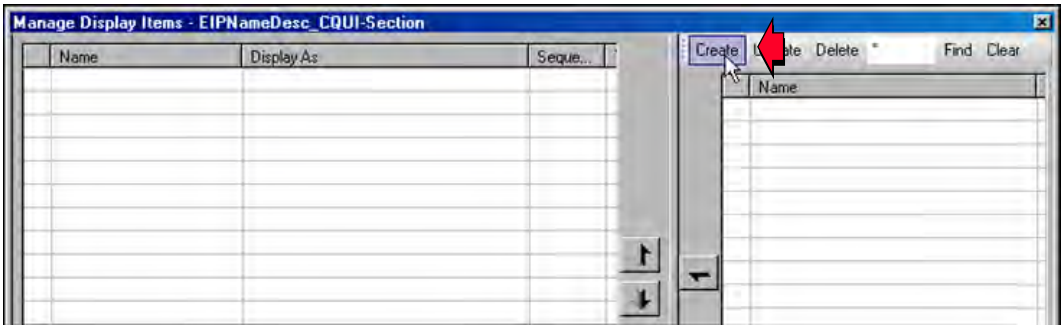


© 2008. Intergraph Corp.
All Rights Reserved.

The display items in a section can be created through the *Manage Display Items* form.

Creating a Display Item Manually

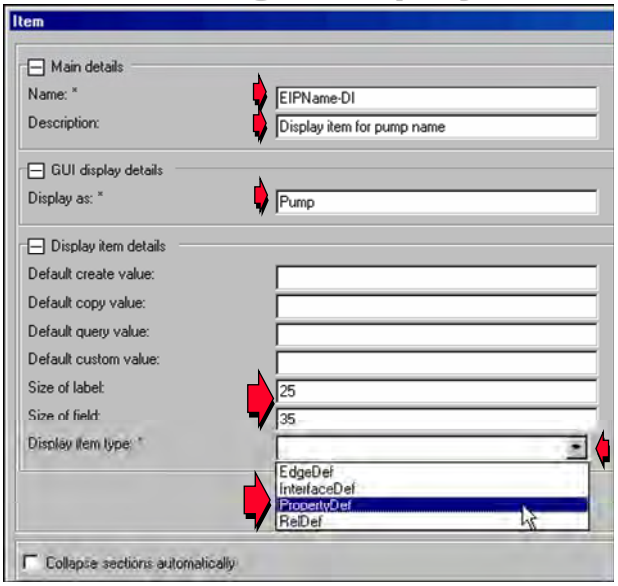
- **Select *Create* from the menu to define a new section display item.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Item* form will appear.

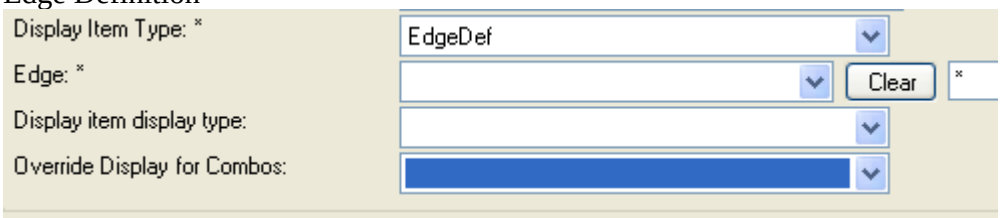
Creating a Display Item Manually



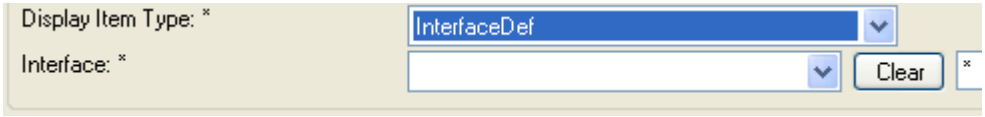
- **Enter the information needed to create a new display item.**
- **Choose the *Display item type* from the picklist.**

The *Item* form (for a display item) contains the following fields:

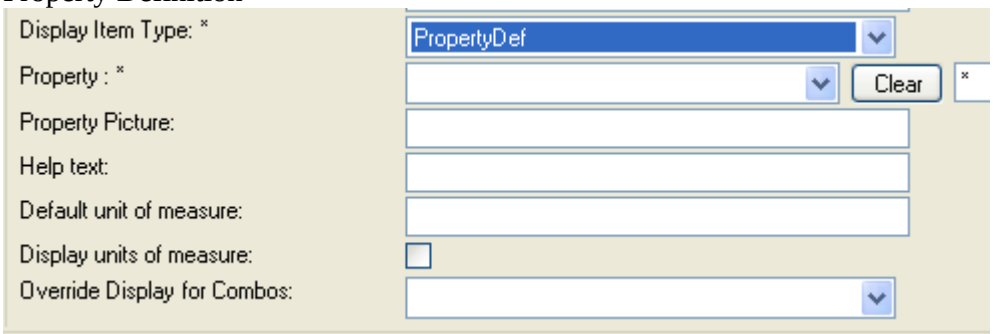
- **Name** - Display item name and is a required field.
- **Description** - Display item description.
- **Display as** - The name the item will display on the form and is a required field.
- **Default create value** - The default value for the display item in *New* dialog boxes that contain this display item.
- **Default Copy Value** - The default value for the display item in *Copy* dialog boxes that contain this display item.
- **Default Query Value** - The default value for the display item in *Query* dialog boxes that contain this display item.
- **Default Custom Value** - The default value to use on special forms.
- **Size Of Label** - The length of the display name.
- **Size Of Field** - The input field size. (The input is truncated to the specified size before it is stored.)
- **Display item display type** - Select the type of item to be displayed. Making a selection in this box will determine any additional fields to be displayed. The options are:
  - Edge Definition



  - Interface Definition

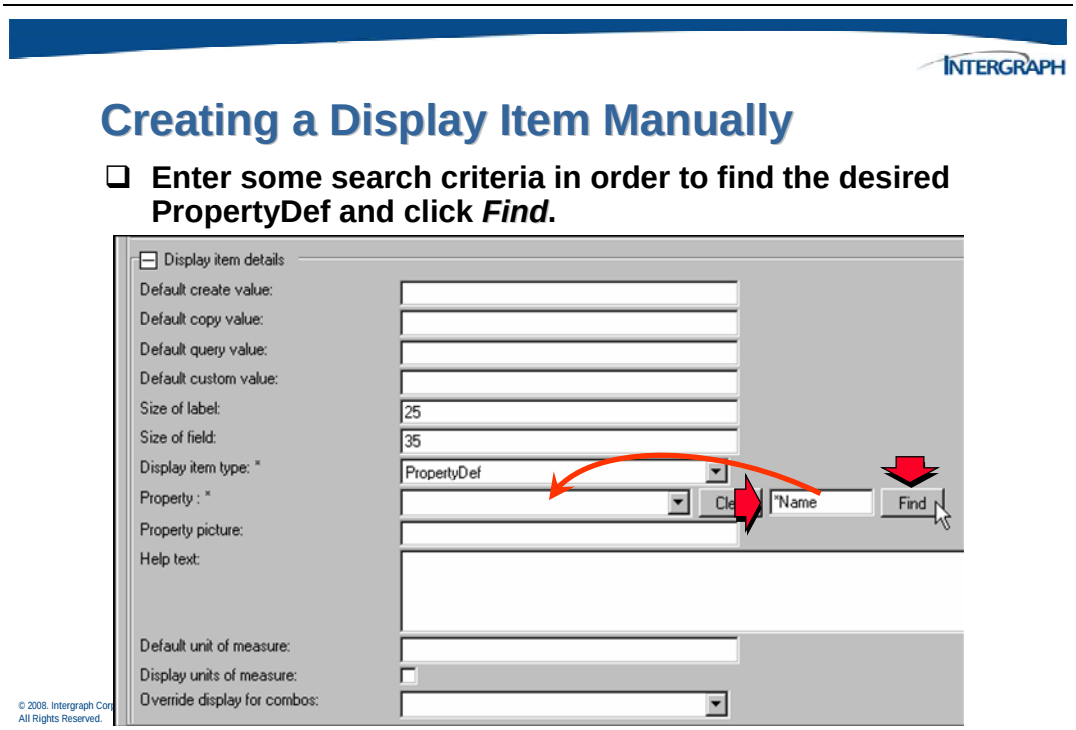


  - Property Definition



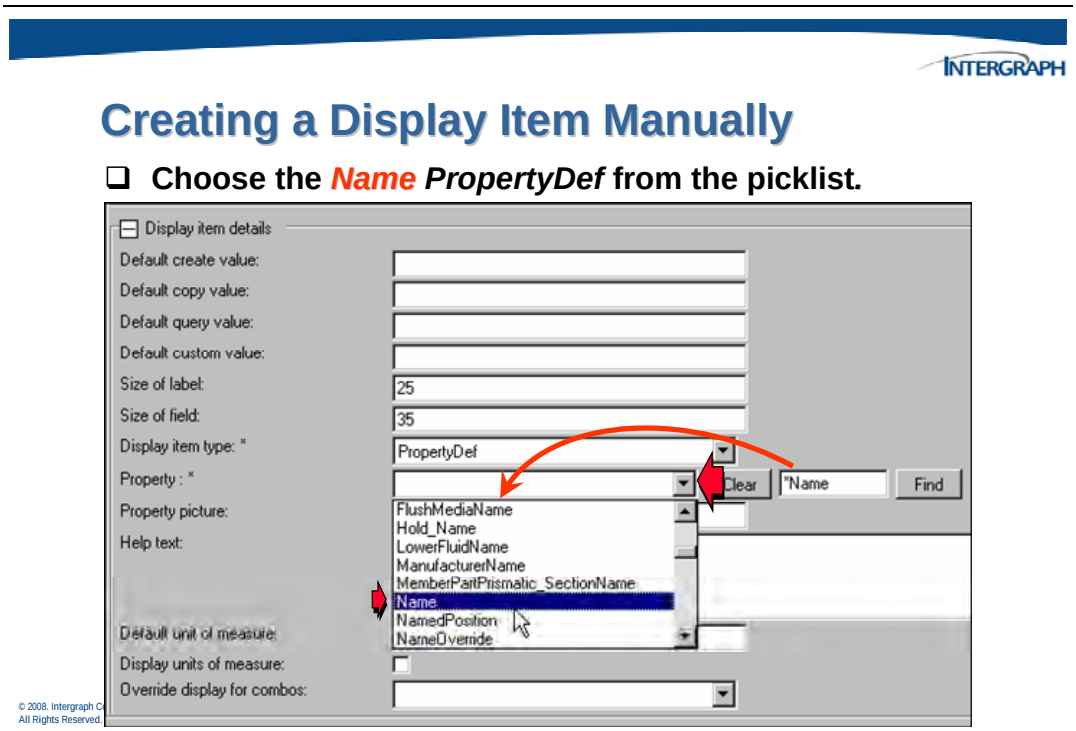
- Relationship Definition

To add the *Name* property as a display item, use the ***Find*** button.

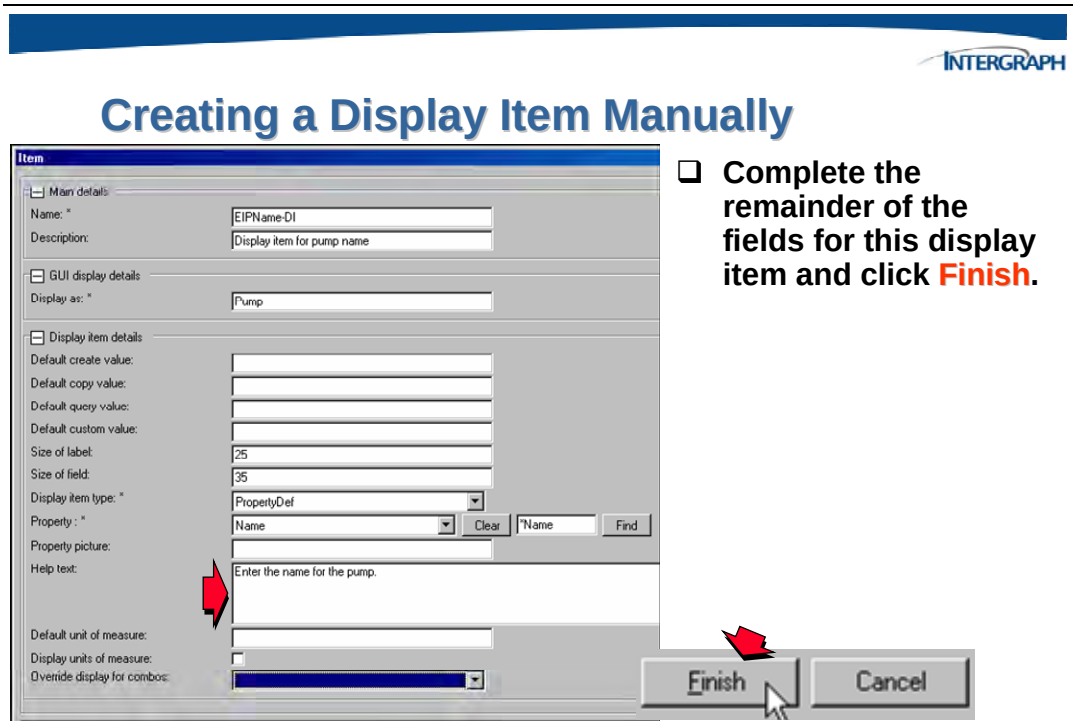


The results of the search will not automatically display in the *Property* field.

Use the list button to see the search results.



This will set the *Property* field to **Name** (IObject.Name).



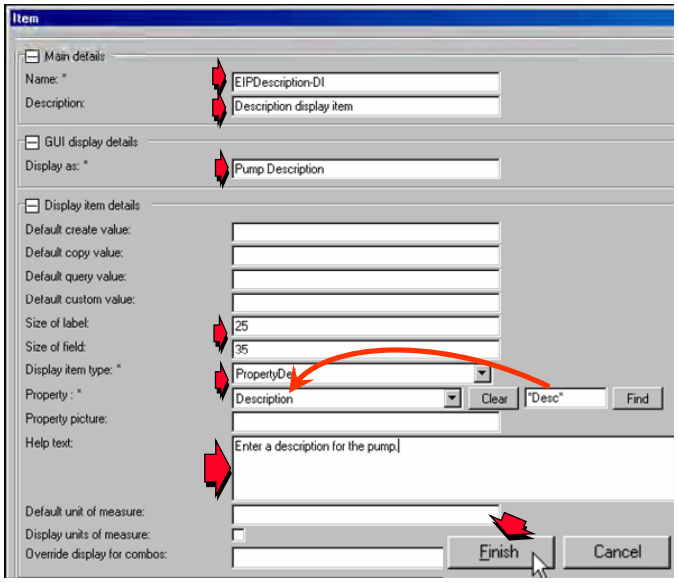
The *Item* form fields con't:

- **Property Picture** - A format for the display item data. The format can either be U for uppercase or any picture mask as defined in Picture Mask Types.
- **Help text** - Help text for display item. (The text to appear when the mouse is hovered over the display item.)

Add another display item, *Description*, to the section.

Creating a Display Item Manually

- **Repeat the process to create a "Description" display item and click Finish.**

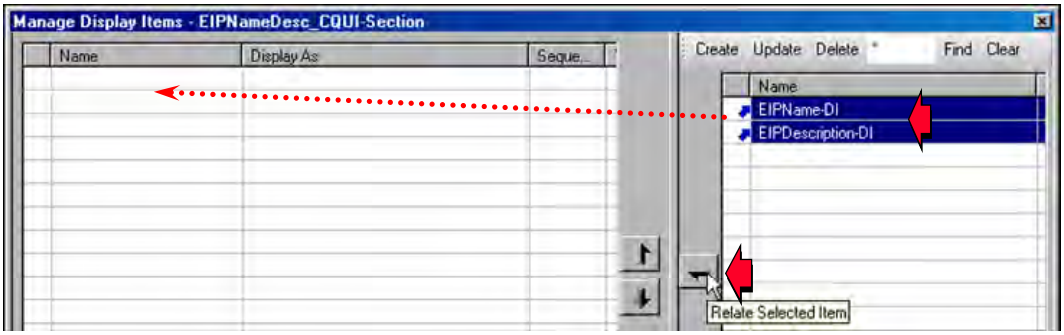


© 2008. Intergraph Corp.
All Rights Reserved.

The new display items are displayed in the *Library* pane.

Creating a Display Item Manually

- **Select *EIPName-DI* and *EIPDescription-DI*, click the left arrow button to relate the display items to the section .**

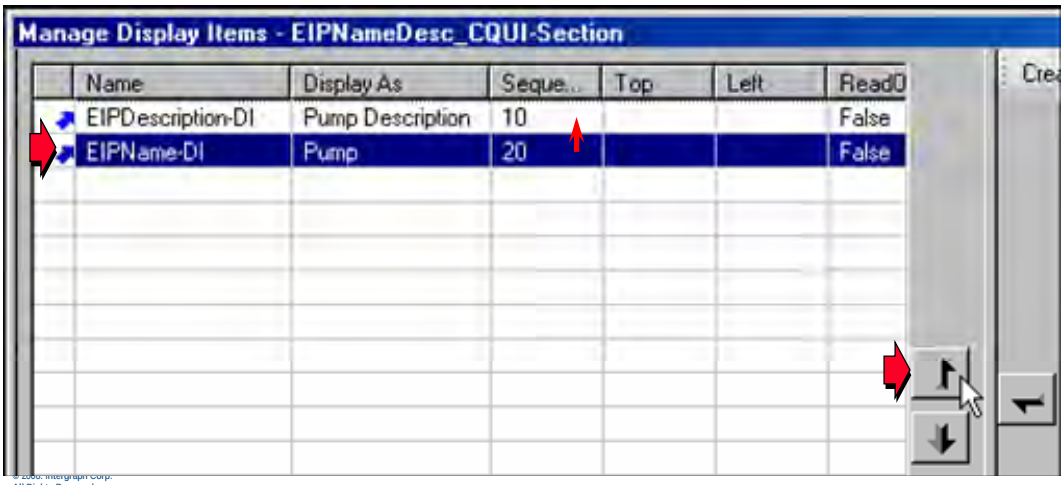


© 2008. Intergraph Corp.
All Rights Reserved.

The new display items are related to the form section and appear in the *Related Items* pane.

Creating a Display Item Manually

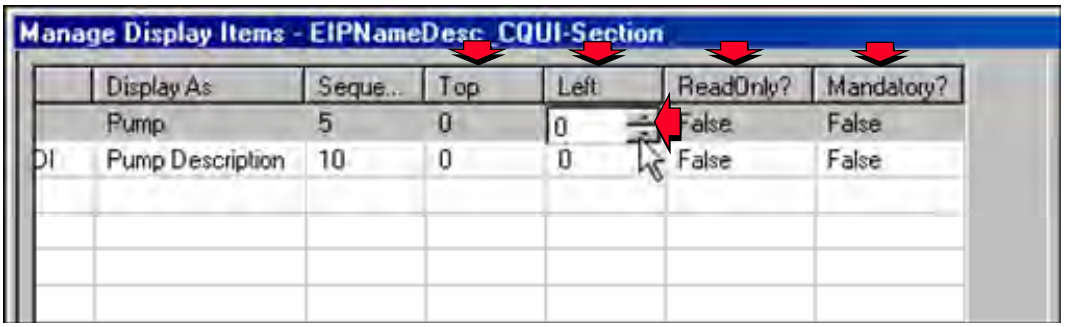
- **Use the up arrow to change the sequence number of the display items.**



All Rights Reserved.

Creating a Display Item Manually

- **Click and set the relationship properties; *Top*, *Left*, *ReadOnly?*, *Mandatory*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Related Items* pane contains the following columns:

- **Name** - Display item name.
- **Display As** - The display item name as displayed on the form.
- **Sequence** - The display sequence of the item on the section.
- **Top** - The number of rows between this display item and the beginning of the section. By default, the top position is set to 0.
- **Left** - The number of columns from the left of the form that the display item should be indented. By default, the left position is set to 0.

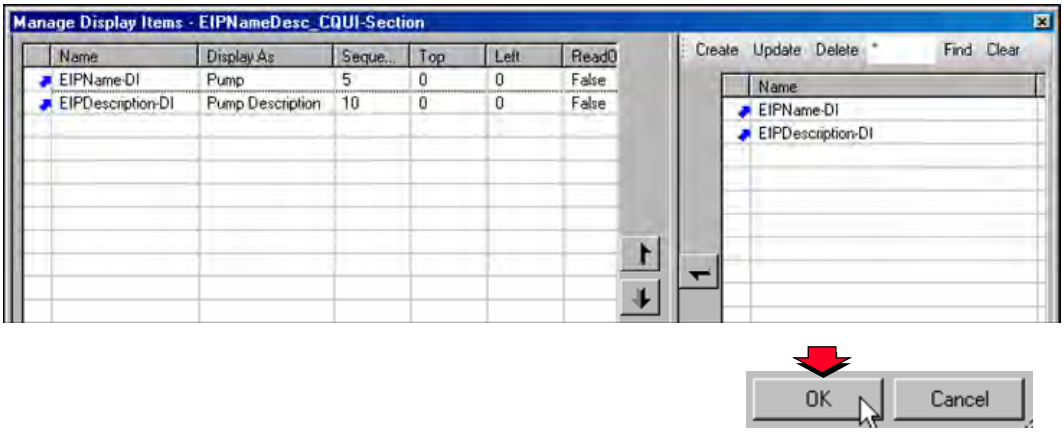
  For example, if you want your section to contain two columns of display items, you could give the first item in the first column a left position of 0 and the first item in the second column a left position of 50.
- **ReadOnly?** - The display item is read only. (If not selected, access is Read/Write)

  The user access specified here can be overridden if attribute-level user access is defined for the display item when it was created.

  The form automatically sets the display items to read-only mode if the purpose of the section in the form is set to *Get Information*.
- **Mandatory** - Display item contains required values that must be entered during create or update operations.

Creating a Display Item Manually

- **To save the new display item relationships, click OK.**

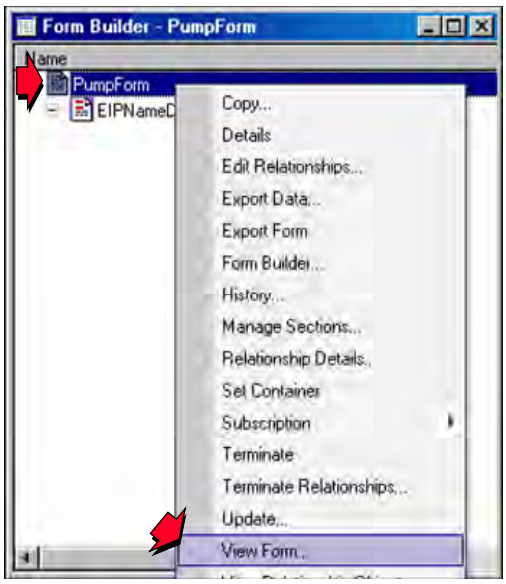


© 2008. Intergraph Corp.
All Rights Reserved.

After the new sections and display items have been created, view the form to verify the appearance.

Viewing a Form

- **Right click on *PumpForm* and from the shortcut menu, select *View Form*.**

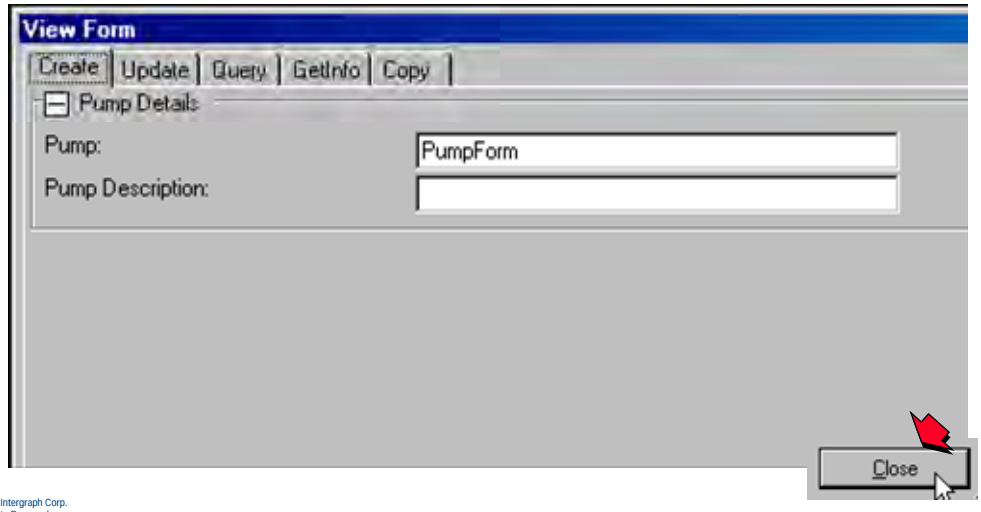


© 2008. Intergraph Corp.
All Rights Reserved.

The *View Form* form will appear.

Viewing a Form

- **Click Close to dismiss the form.**

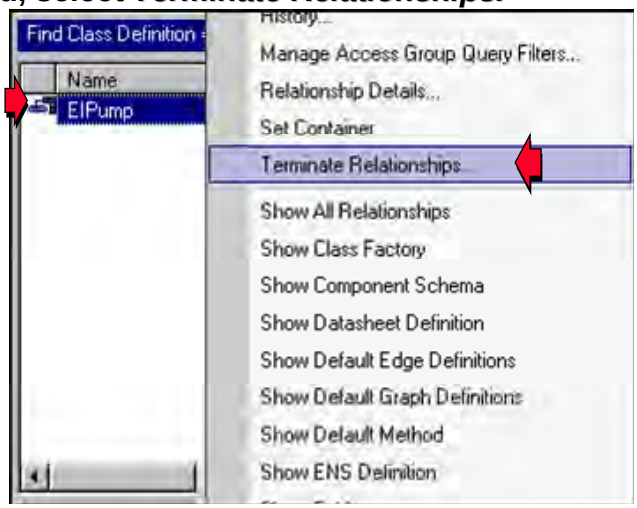


© 2008. Intergraph Corp.
All Rights Reserved.

Since the **EIPump** class already has an associated form, the relationship needs to be terminated so that the new form can be related.

Removing Old Form Relationships

- **Right click on the *EIPump* class and from the shortcut menu, select *Terminate Relationships.***

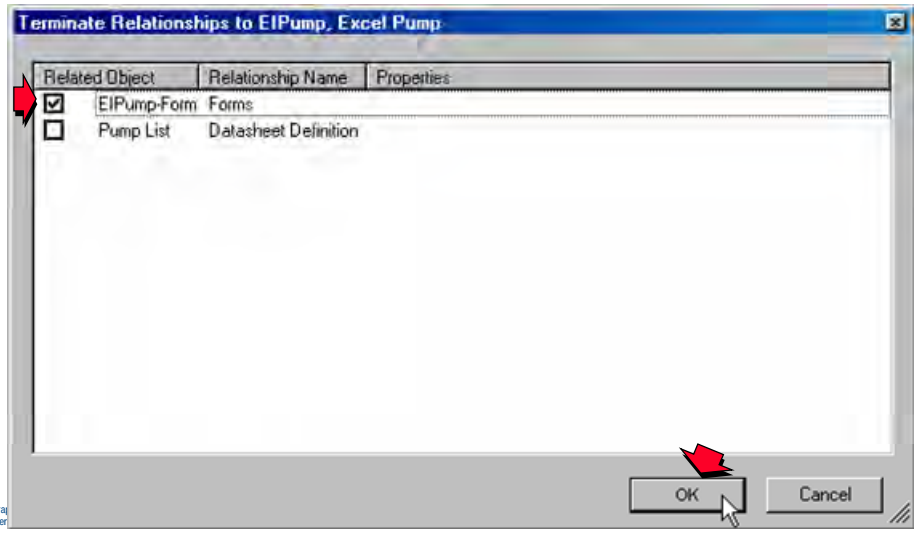


© 2008. Intergraph Corp.
All Rights Reserved.

The *Terminate Relationships* form will appear.

Removing Old Form Relationships

- **Select the relationship to terminate and click OK.**

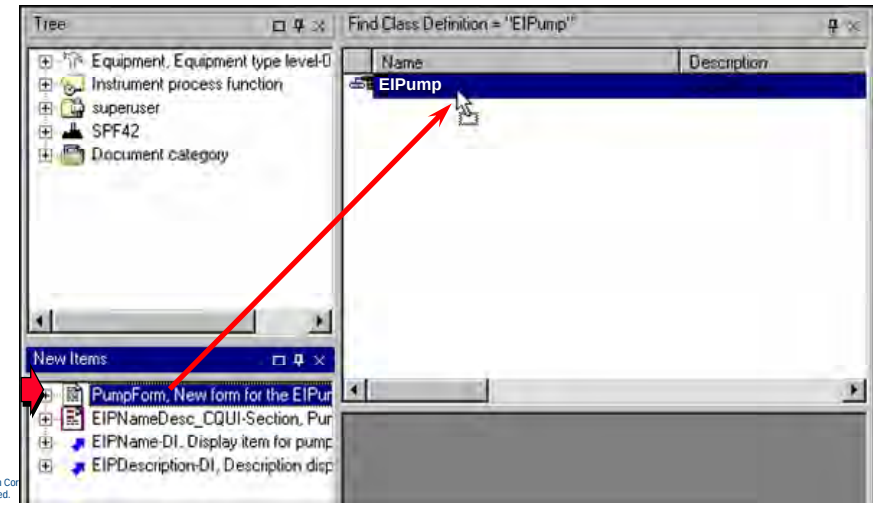


© 2008. Intergraph
All Rights Reser

Relate the new form to a class definition using drag and drop.

Relating a New Form

- **Drag and drop the new *PumpForm* onto the *ElPump* ClassDef.**

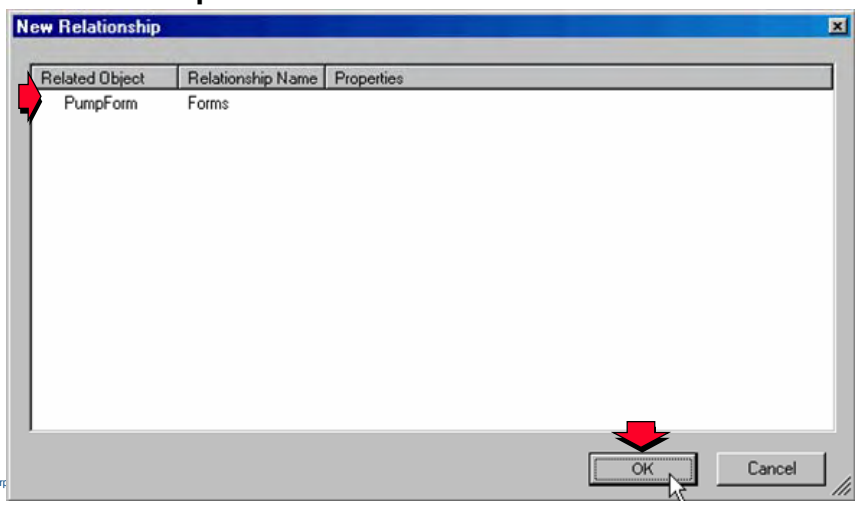


© 2008. Intergraph Cor
All Rights Reserved.

The *New Relationship* form will appear.

Relating a New Form

- **Click Finish to create the new Forms (Form➔ ClassDef) relationship.**

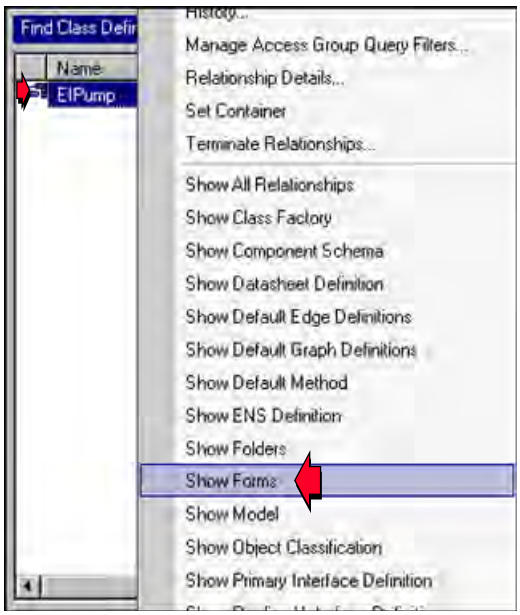


© 2008. Intergraph Corp.
All Rights Reserved.

Verify that the new form/class relationship was created.

Relating a New Form

- **Right click on the *ElPump* class and select *Show Forms* from the shortcut menu.**

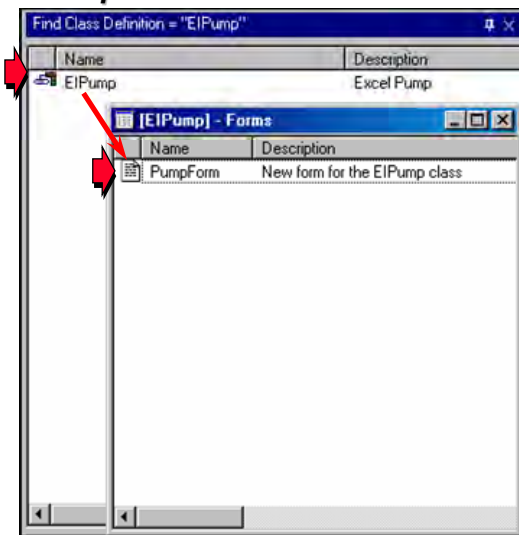


© 2008. Intergraph Corp.
All Rights Reserved.

Another list view window will be opened to show the related forms.

Relating a New Form

- **Verify the *PumpForm* is related to the *EIPump* class.**

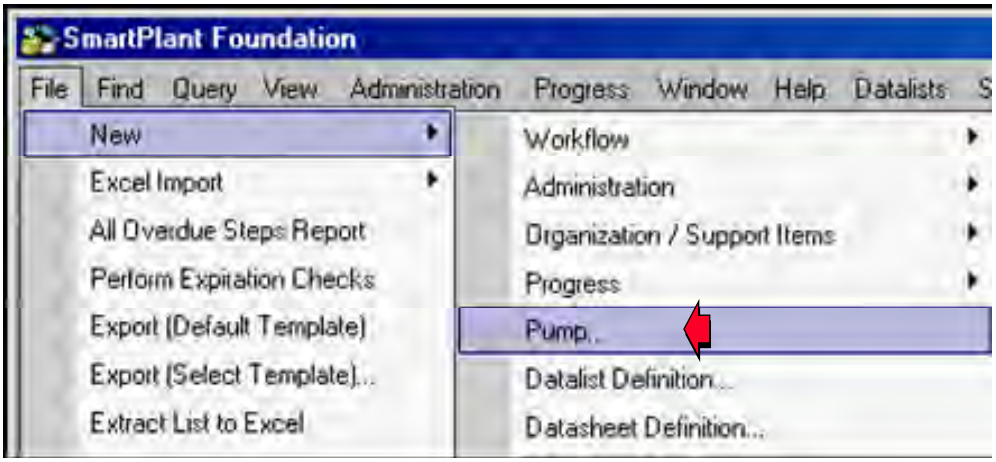


© 2008. Intergraph Corp.
All Rights Reserved.

Use the create pump method (EIPumpCreate) to test the new form.

Testing the Form

- **From the menu, select *File* > *New* > *Pump* in order to display the new pump form.**

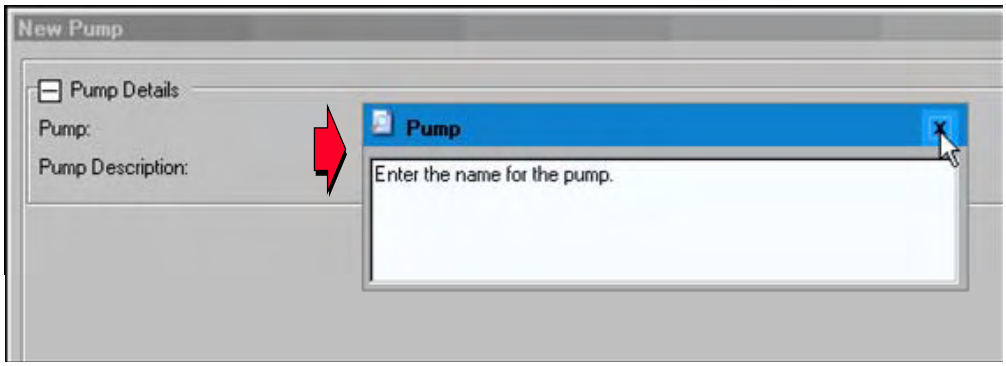


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Pump* form will appear.

Testing the Form

- **Hover the cursor over the *Pump* (name) field to see the Help Text displayed.**



© 2008. Intergraph Corp.
All Rights Reserved.

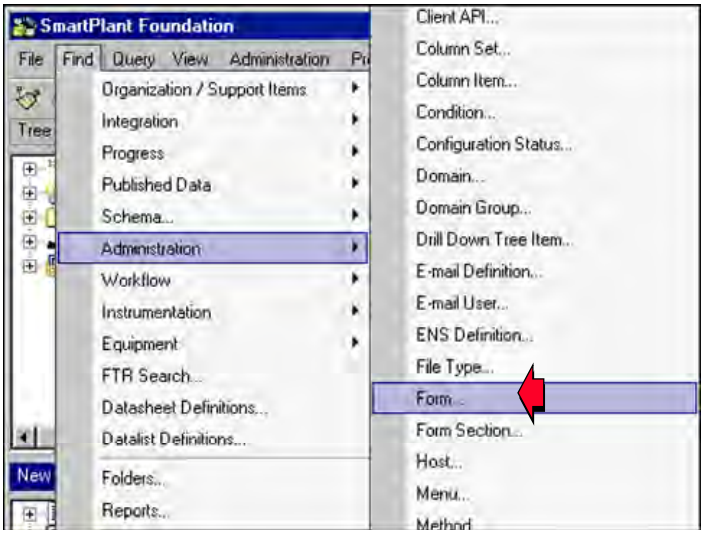
4.4.3 Copying Forms

Copying a form allows you to create a new form definition that references the same sections and display items as the original form. It also copies all the associated relationships.

Begin by performing a find to locate an existing form.

Copying a Form

- **Select *Find* > *Administration* > *Form* from the menu.**

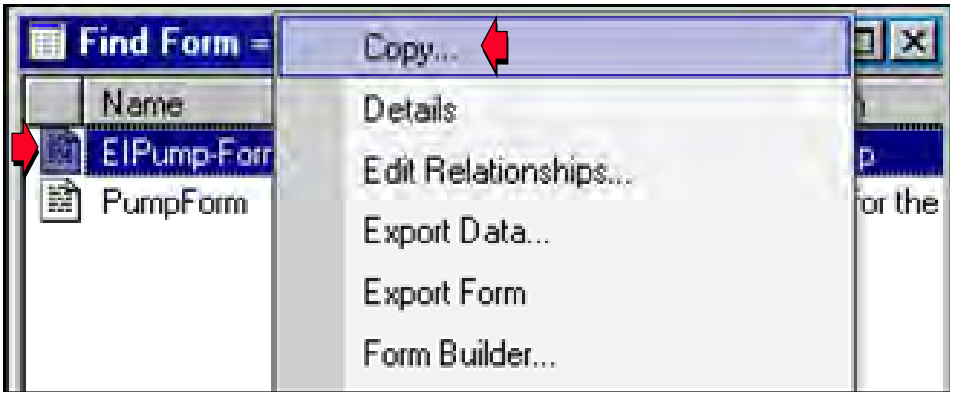


© 2008. Intergraph Corp.
All Rights Reserved.

From the list view window, select the form to be copied.

Copying a Form

- **Right click on the *ElPump-Form* and from the shortcut menu, select *Copy*.**

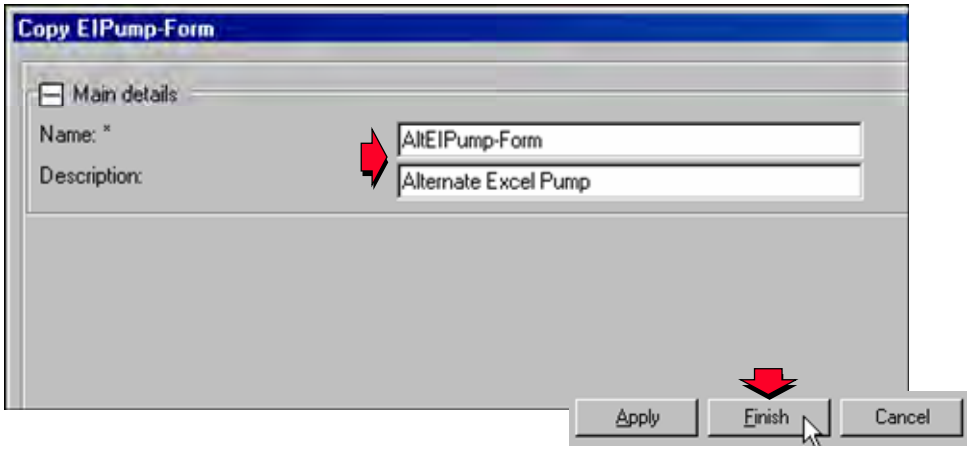


© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Form* form will appear.

Copying a Form

- **Enter a new required *Name* and optional *Description* for the form copy and click Finish.**



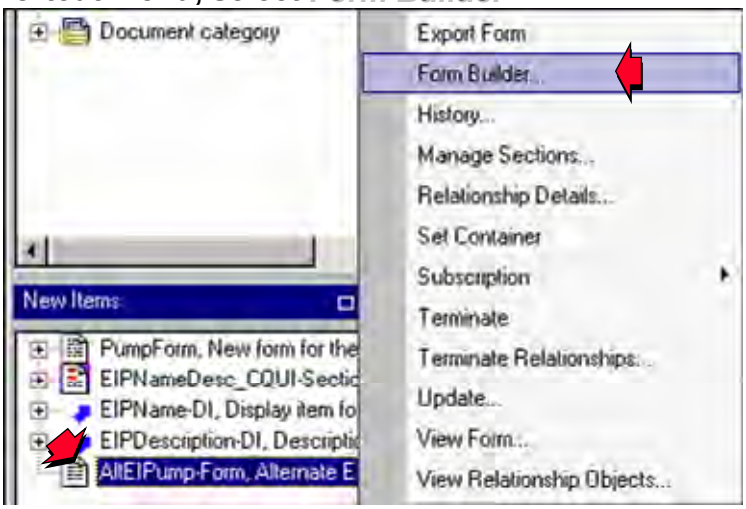
© 2008. Intergraph Corp.
All Rights Reserved.

When the form is copied, the relationship to the class definition is also copied.

Use **Form Builder** to create or edit the form sections and display items.

Copying a Form

- **Right click on the *AltElPump-Form* and from the shortcut menu, select *Form Builder*.**

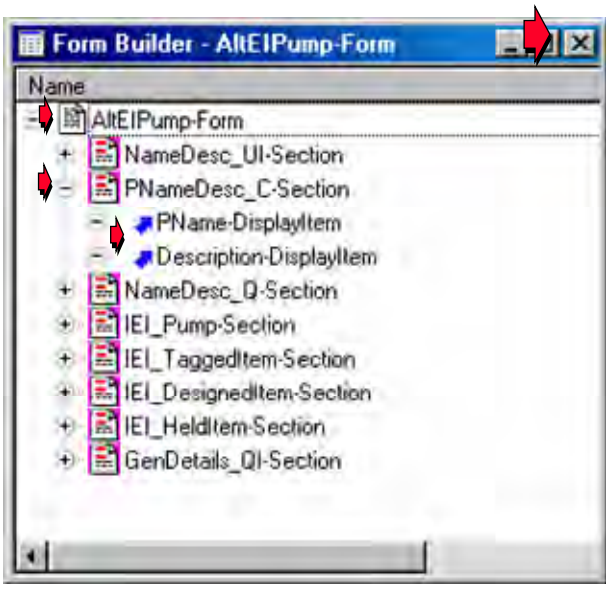


© 2008. Intergraph Corp.
All Rights Reserved.

The *Form Builder* window will show the **form**, related **sections** and related **display items**.

Copying a Form

- **Expand the *AltElPump* form and the *PNameDesc_C* section to view the form contents.**



© 2008. Intergraph Corp.
All Rights Reserved.

4.5 Activity 2 – Manually Creating Forms

Complete **Chapter 4** - **Activity 2** in the SPF Configuration and Administration II activity workbook.

CHAPTER 5

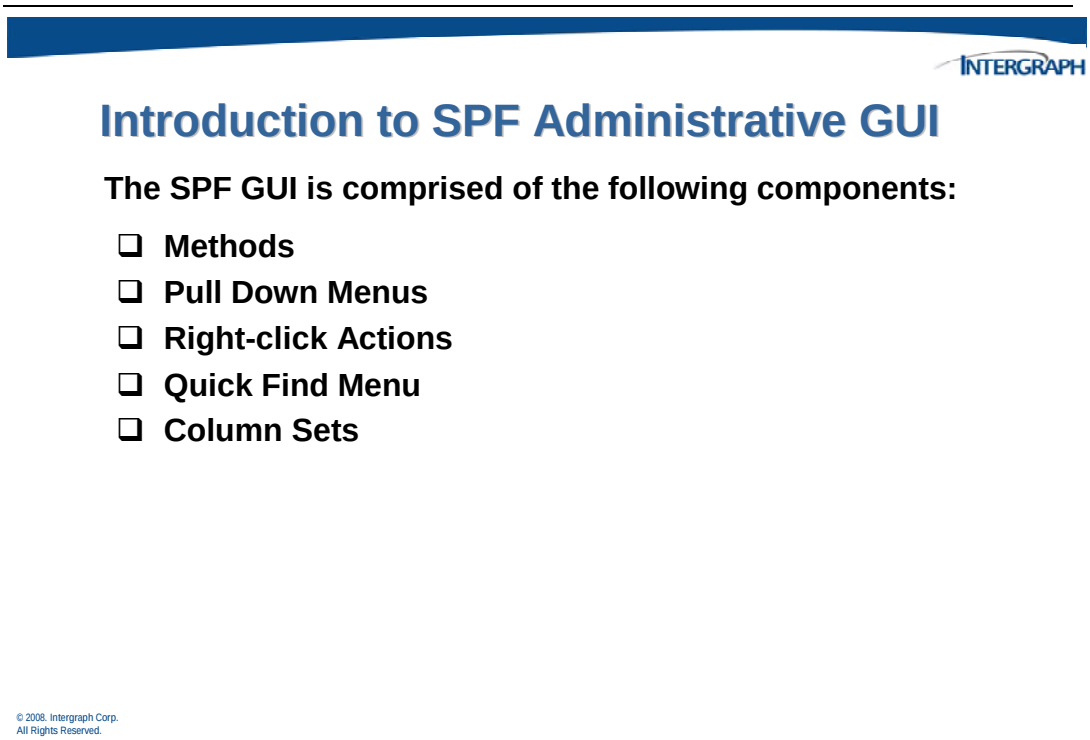
Configuring Methods, Menus and Toolbars

5. The SPF Administrative GUI

This chapter will cover the details of how to configure the Graphical User Interface (GUI) in SmartPlant®Foundation using menus, methods and toolbars.

Upon exiting SPF, the Desktop Client saves a user profile that contains information about the windows, configuration, and other user interface settings. The user profile saves positional information for the tree view, the new items window, and the properties grid, as well as the display settings for the windows.

When the software is started again, the Desktop Client user interface displays these windows in the same locations they were when the user exited. However, the user profile does not save list view windows.



The following gives a brief description of the GUI components that will be covered in detail in this chapter

Methods

In SmartPlant Foundation, information is retrieved by performing certain operations on interfaces of objects. The availability of these operations is controlled by assigning methods to interface definitions. Users have access to these operations by selecting commands from the main menus or a shortcut menu via the right hand mouse (RHM).

Right-Click Actions

Methods can be attached to an object and exposed by a right-click (also called right-hand mouse or RHM).

There are two types of shortcut menu commands:

- **Actions** - for example, *Export Data, Manage Related Items*
- **Relationships** - for example, *Show Folders, Show Display Items*

Pull Down Menus

Pull down menus are used to group various actions performed in the Desktop Client. Menu items can be created for each menu specifying the action to be performed, the method to be used, and other related properties. Menus can also contain sub menus.

Menus are static and displayed regardless of the user group.

Notes:

- If the menu is empty, it will not be displayed.
- The menu nodes and related menu items can be assigned user group access. A user must be in a role with access to the user group associated with the menu in order to gain access to that menu item.

Toolbars

Toolbar items provide a shortcut for performing various actions in the Desktop Client. These items (also called toolbars in SmartPlant Foundation) can be created and added to the main toolbar.

The definition for each toolbar button contains the action to be performed the button is clicked, the method to be used for the button, and other related properties.

User group access can be assigned to the toolbar item. A user must be in a role with access to the user group associated with the toolbar in order to access to that toolbar item.

Quick Find Menu

The SmartPlant Foundation tool bar includes a Quick Find (Qfind) menu that lists all the Quick find methods that a user has access to through their role permissions.

Users can select a method from an items list (left box) or enter all or part of the object's name in the Find box (wildcards (*) can be used).

The predetermined Quick Find items delivered with the core model include:

Name	**Description**
DesignDocumentQFind	Design Documents
DevSystemFind	Systems (DEV)
DEVTagFind	Find Tags (DEV)
SPFVendorActiceDocFind	Vendor Active Documents QFind
SPFVendorAllDocFind	Vendor All Documents Qfind
SPFVendorReservedDocFind	Vendor Reserved Documents Qfind

Relationship Management

One way relationships can be created in the Desktop Client is through 'drag and drop'. Multiple disparate objects can be dropped onto a single object to create a disparate set of relationships.

The relationship management commands are run from the shortcut menu command of an object. Including:

- View relationships
- View related objects
- Edit relationships
- Delete or terminate relationships

Manage Related Items

Manage Related Items is a new shortcut menu command available on a range of objects. The command displays a standard form that can be used to easily find, create, order, update, and delete items related to the object.

Column Sets

Column sets determine how objects are displayed in the Desktop Client (list views) and manage related items support.

Column sets are collections of column items, which can be used to display edge definitions, property definitions, and relationship definitions.

5.1 Introduction to API's and Methods

In the SmartPlant Foundation system, information is retrieved by performing certain operations on Interface Definitions. The availability of these operations is controlled by assigning methods to the Interface Definitions. Users have access to these operations by selecting them from the main menus or the shortcut menu.

Method behavior is driven by client APIs; the method arguments defined on the API are dynamic and set on a form along with some additional properties.

When a method is configured, the new method can be attached to either a component or a Client API.

Functionality Configuration

- **End user functionality is configured by using methods that are based on Client APIs.**
- **Methods can be attached to Interface Definition where they are available only when the Interface Definition instance is in focus or to menu items/toolbars when they are always available.**
- **Access to functionality can be controlled by Access Groups, Owning Groups and conditions.**
- **Access Groups are associated to methods by using manage related items.**

© 2008. Intergraph Corp.
All Rights Reserved.

5.1.1 Client API's

Client APIs detail the methods allowed in SmartPlant Foundation to support the server assemblies installed. A set of Client APIs are delivered during the software installation.

INTERGRAPH

Client APIs

Client APIs are the actual functions that can be used in methods to expose system functionality for the end users

A Client API can be used in several methods

- **Client API name**
- **Client API description**
- **Necessary parameters**

© 2008. Intergraph Corp.
All Rights Reserved.

Client APIs

These are some of the existing client APIs delivered with SPF.

Find Client API = "*"

Name	Description	Parameter 1	Parameter 2	Parameter 3
Activate	Activate a reserved object	Present the classification tree	Classification tree filtering	Form purpose name
AddOptionalIFaces	Admin function will display a GUI allowing ...			
AllOverdueStepsReport	All Overdue Steps Report	ViewDef name	Output type	Template name
ApplySchemaChanges	Apply schema changes to the database			
ArchivePurge	Archive and Purge selected files	GraphDef for structure	Target location	Archive
AttachWorkflow	Attaches and Item to a workflow	Allow edit after attach		
BatchPrint	To print file(s)			
ChangeMouseClickAction	To set default mouse click action on object			
ChangeOwnerOwningGroup	Change Owner and Owning Group	All Owning Groups?		
ChangeOwningGroup	Change Owning Group	All Owning Groups?		
ChangeStatus	Create Status	From state	To state	Cleanup Subscription
ChangeUser	Change to another user			
ChangeUserPassword	Change User Password			
CheckIn	Function to Check in files to the vault	Replace	Reference file not found action	Get checked out ver
CheckOut	Function to Check Out files from the vault	Show hidden files		Checkout top files
ClaimObj	Claim Object	Column set to present results	Auto Subscribe to Claimed Event	Auto Subscribe to Ne
ClearTree	Clear Tree			
CompareDBasis	Compare Design Basis	Tool Comp Schema	Tool Domain	Adapter Code
CompareLatestRetrievedData	Compare Latest Retrieved Data	Tool Component	Tool Domain	Adapter Code
CompareObjsToHigherConfig	Compare Objs To Higher Config	Mode	Column set to present results	
ConfigRoleAssignment	Launch Config Role Assignment			
ConfigurationReport	Configuration Report	ViewDef name	Output type	Template name
ConfigureDatalist	Launch the SPF datalist criteria form.			
ConvertForNavigation	Creates a navigation file			
CopyNamedObject	Transmittal from template			
CopyObj	Copy object and relations	Title	Copy files	Show hidden files

© 2008. Intergraph
All Rights Reserve

5.1.2 Conditions

Conditions can be used to control the availability of methods on the shortcut menu and pull down menus and to control when it is valid to create a relationship between two objects. These two situations separate the conditions into two basic types:

- Object conditions which operate on a single object.
- Relationship conditions which operate on the two objects participating in a relationship and occasionally the relationship itself.

It is also possible to use the environment object that contains properties about the current users session such as the user name and access groups.

INTERGRAPH

Conditions

Conditions can be created against any property or objects participating in a relationship.

Conditions consist of the following:

- **Condition name**
- **Condition description**
- **Condition default (Obj.Property=value or Obj -> RelDef_12)**

© 2008. Intergraph Corp.
All Rights Reserved.

Condition Syntax

To access the primary objects:

The syntax for accessing the object in the object condition is:

- Obj - for the main object

The syntax for accessing these objects is:

- Obj - for the relationship object
- Obj1 - for the object 1 (as defined on the RelDef)
- Obj2 - for the object 2 (as defined on the RelDef)

The syntax for accessing the environment object is:

- Env - for the environment object

To access an object property:

Properties on an object are referenced by name after the key word shown above with a '.' delimiter. For example to reference an object's name use:

- Obj.Name

To access a related object:

Related objects can be referenced by *RelDef* or *EdgeDef* name after the key word shown above with the '→' (->) delimiter.

When using a *RelDef*, the '_12' or '_21' direction qualifier has to be appended to the RelDef name. For example to reference an objects owner by RelDef use:

- Obj→SPFItemOwner_12

When using an EdgeDef, the '**EDG_**' direction qualifier has to be prefixed to the EdgeDef name. For example to reference the latest version of a document master by an EdgeDef use:

- Obj→EDG_LatestVersion

To access a property on the related object treat it in the same way as the main object:

- Obj→SPFRevisionVersions_21.SPFRevState

Only one relationship "hop" can be navigated from each of the main objects.

Simple Comparisons:

A condition is made up of two expressions and a comparison operator. The following comparison operators are supported:

- =
- <>
- like (use * for wild cards)
- < and <=
- and >=

The following Boolean operators are supported:

- And
- Or
- ( )

It is not possible to mix the *And* and *Or* operators in a single condition.

For example:

obj.SPFPrgActive=True and obj.SPFPrgDeliverablesFrozen=True

IN List Syntax:

To test if one value is in a list of string:

TestString IN ('String1','String2',…)

For example:

Obj->SPFFileFileType_12.UID IN
('FT_AcadFile','FT_UstnFile','FT_DXFFile','FT_IGRFile','FT_PIDFile',
'FT_SHAFile','FT_SPEFile','FT_ZyQFile')

Listing and Objects Interfaces:

The following syntax returns a comma separated string of the interface names instantiated by an object:

Obj.Interfaces

String Comparison:

Instead of using the comparison for operators shown above, the following syntax can be used to test the existence of one sub string in another:

INST(MainString,SubString)

Logical expressions that evaluate to a string can be used for either string.

INSTR(Obj.Interfaces, 'ISPFTransmittal')

Counting the Number of Related Objects:

The following syntax returns an integer:

Count(Obj->RelDef_12)

Example - checking if an object has an owner:

Count(Obj->SPFItemOwner_12)=1

The following functions can be used when creating condition expressions:

Function Name	Description	Example
ABS	Specifies the absolute value of a number.	ABS(Obj.Value1) > 3
AVG	Specifies the average value for a returned set of values.	AVG(Obj.Value1, obj.Value2) < 5
COUNT	Counts the number of objects in a set.	COUNT(Obj->TagArea)

		> 0
INSTR	Searches for a particular string in the values returned for the attribute.	INSTR(Obj.Name, "Admin User") > 0
LEFT *	Specifies characters on the left side of a string.	LEFT(Obj.Name) = "ABC"
LEN	Counts the number of characters in a string.	LEN(Obj.Name) > 5
LOWER	Converts the contents of the returned string to lowercase characters and compares the results to the specified string.	LOWER(Obj.Name) = "graph"
MAX	Specifies the highest of the returned values.	MAX(Obj.Value1, Obj.Value2) > 5
MID *	Returns all characters from a given position in a string to the end of the string.	MID("Intergraph", 6) would return a value of "graph".
MIN	Specifies a minimum value for returned values.	MIN(Obj.Value1, Obj.Value2) < 3
RIGHT *	Specifies characters on the right side of a string.	RIGHT(Obj.Name) = "XYZ"
ROUND	Rounds the returned number to the nearest integer.	ROUND(Obj.Value1) >= 3
SUM	Evaluates the sum of all returned values.	SUM(Obj.Value1, Obj.Value2) < 5
TRUNC	Specifies the integer portion of a number.	TRUNC(Obj.Value1)=3
UPPER	Converts the contents of the returned string to uppercase characters and compares the results to the specified string.	UPPER(Obj.Name) = "ADMIN"

* These functions currently have a TR outstanding on them as they were accidentally commented out when the code was ported to 4.x from 3.x. These functions are planned to be re-instated in service pack 2.

Environment Variable Support

Conditions Environment object properties and their meanings (where required)

HOSTNAME	Current SPFServer machine name
USERNAME	Current user's user name
USER	Exposes the currently logged in ISPFLoginUser object to support access to additional properties on this object.
USERORGANIZATIONNAME	Current user company name
USERUSERTYPE	Returns Full, E-mail or Paper based on whether the current user class has the active domain of ISPFLoginUser, ISPFEmailUser or ISPFPaperUser
CONFIGCREATE	Current create config string
CALLEDFROMDIALOG	Returns VBClient/WebClient based on a session parameter
CALLEDFROM	As Above
EFFECTIVITYDATE	Current effectivity date
CONFIGLEVEL	Current create config level 0,1,2 etc.
CURRCONFIGLEVELNAME	Same as QUERYCONFIGSTRINGS
CREATECONFIGUID	Current create config UID
CREATECONFIGOBJ	Exposes the current create config object to support access to additional properties on this object.
CREATECONFIGSTRING	Comma separated list of all current create configuration Names
CONFIG1, CONFIG2, ..5	Name of the create config at the specified level
CREATECONFIGLEVELxOBJ	Exposes the create config object at the specified level to support access to additional properties on this object (1..5 supported.).
USERDEFAULTCREATECONFIG	Current user default create config
QUERYCONFIGLEVEL	Current query config level 0,1,2 etc.
QUERYCONFIGUIDS	Comma separated list of all current query configuration UIDs
QUERYCONFIGSTRINGS	Comma separated list of all current query configuration Names
QUERYCONFIG1, QUERYCONFIG2, ..5	Name of the query config at the specified level
USERUSERGROUPS	Comma separated list of usergroups that the current user belongs to. It is better to use the ones below.
OWNINGGROUPSFORCREATECONFIG	All valid owning groups at this config level
OWNINGGROUPSFORQUERY	All valid owning groups at this config level

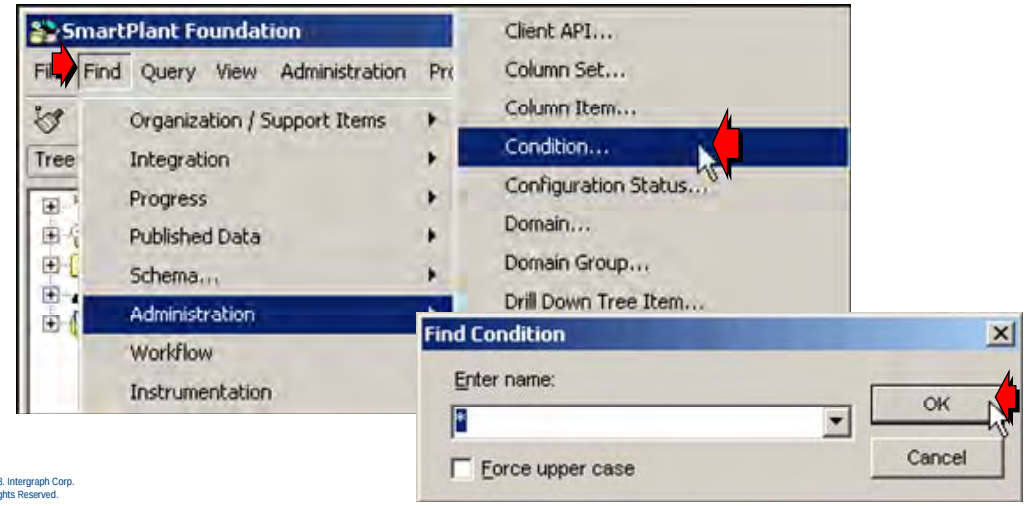
CONFIG	
OWNINGGROUPSFORUSERIN CREATECONFIG	All valid owning groups for this user at the current create config level
OWNINGGROUPSFORUSERIN QUERYCONFIG	All valid owning groups for this user at the current query config level
ACCESSCROUPSFORUSERIN CREATECONFIG	This returns a list of access groups associated with the current create config/user (comma separated string)
ACCESSCROUPSFORUSERIN QUERYONFIG	This returns a list of access groups associated with the current query config/user (comma separated string)

Note: For more detailed information on Conditions, refer to the *How to Configure the GUI Model* document.

SmartPlant Foundation has existing conditions delivered that can be used in the configuration of methods.

Conditions

- **Use the *Find > Administration > Condition* command to locate and view the delivered conditions.**

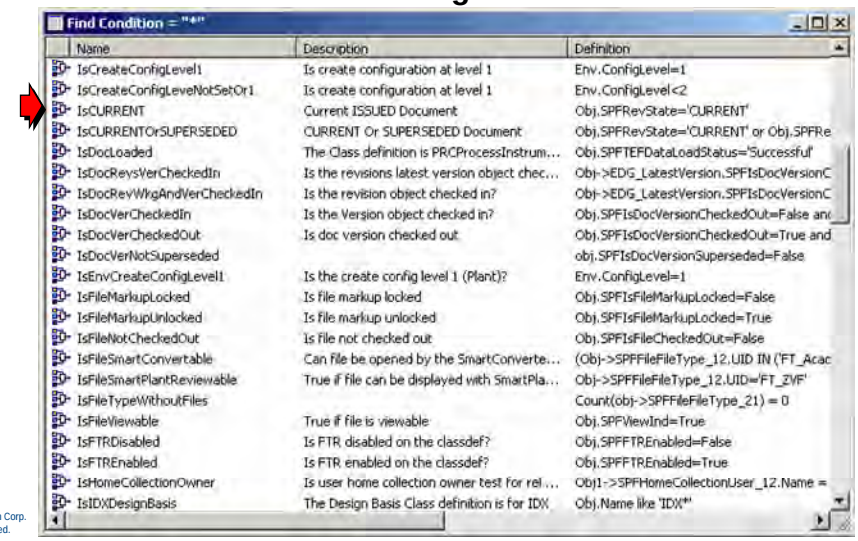


© 2008. Intergraph Corp.
All Rights Reserved.

From the tree, click on the method title to populate the right pane with method properties.

Conditions

These are some of the existing conditions delivered with SPF.

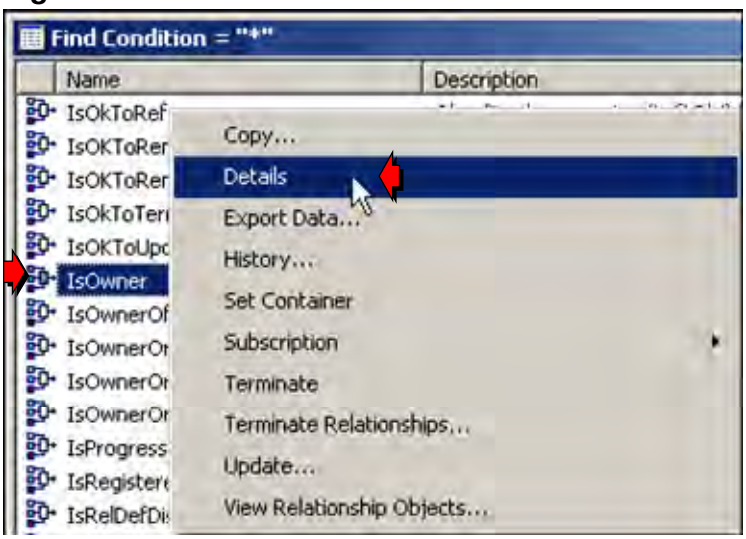


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *Details* command to view the delivered **IsOwner** condition.

Conditions

- **Right-click on the condition IsOwner and select *Details*.**



© 2008. Intergraph Corp.
All Rights Reserved.

Below is an example of the **IsOwner** delivered condition.

Conditions

The properties for the selected condition are displayed in the *Details* form.

© 2008. Intergraph Cor
All Rights Reserved.

5.2 Configuring New Methods

In SmartPlant Foundation, new instances are created by performing a create operation on the object class or information is retrieved by performing certain operations on interfaces of objects. The availability of these operations is controlled by assigning methods to interface definitions. Users have access to these operations by selecting commands from the main menus or a shortcut menu via the right mouse button.

Method behavior is driven by client APIs; the method arguments defined on the API are dynamic and set on a form along with some additional properties.

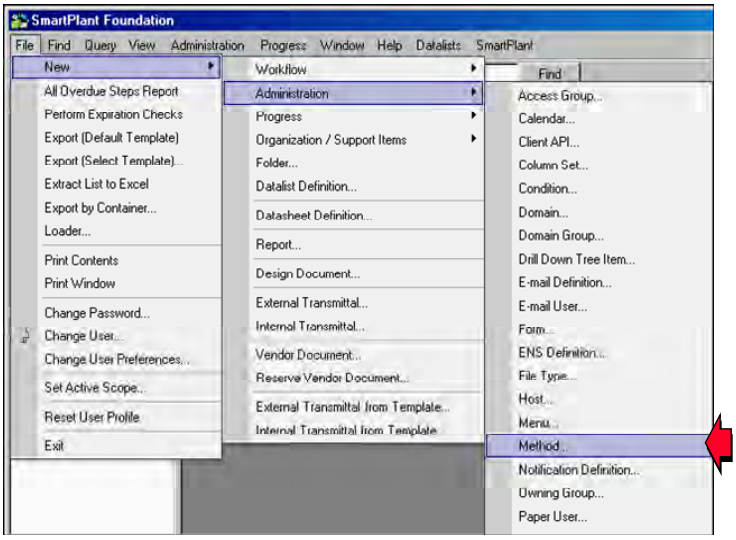
When configuring shortcut menu methods, before creating a new method for a client API, check to see if there is a suitable existing one; that is, one with suitable user access and a condition. If there is, it can be related to the interface to save creating a new method.

To configure a new method:

INTERGRAPH

Creating Methods

- **From the menu, select *New > Administration > Method.***

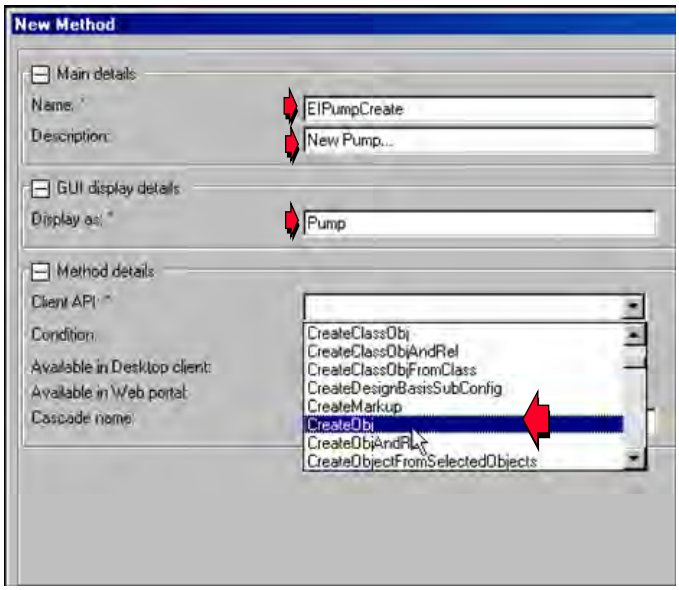


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Method* form is displayed.

Creating Methods

- **Enter the necessary values to define the new method.**
- **Select *CreateObj* for the *Client API*.**



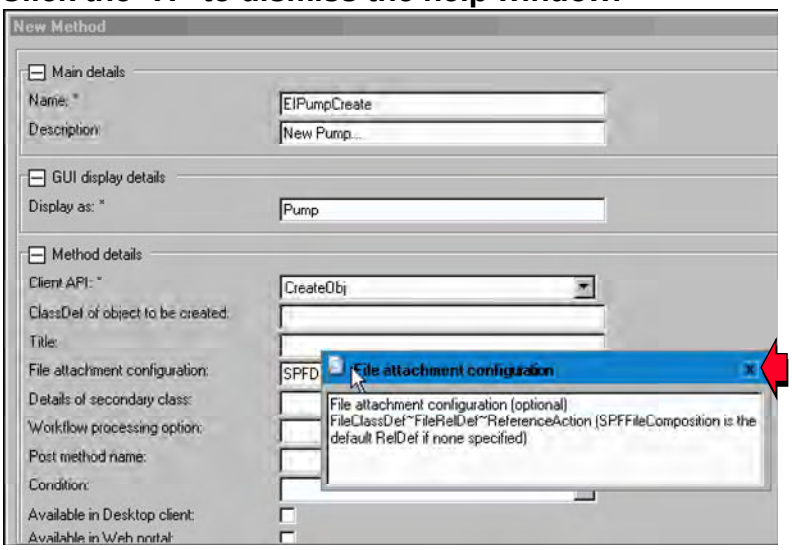
© 2008. Intergraph Corp.
All Rights Reserved.

The *New Method* form contains the following fields:

- **Name** - specifies the name/purpose of the method and is a required field.
- **Description** - specifies a description for the method and is an optional field.
- **Display as** - specifies the shortcut menu display text. This is a required field.
- **Client API** - associates a *Client API* with the method. Select the API from the list. The form changes dynamically to present the configured API arguments. This is a required field.

Creating Methods

- **Click the "*X*" to dismiss the help window.**



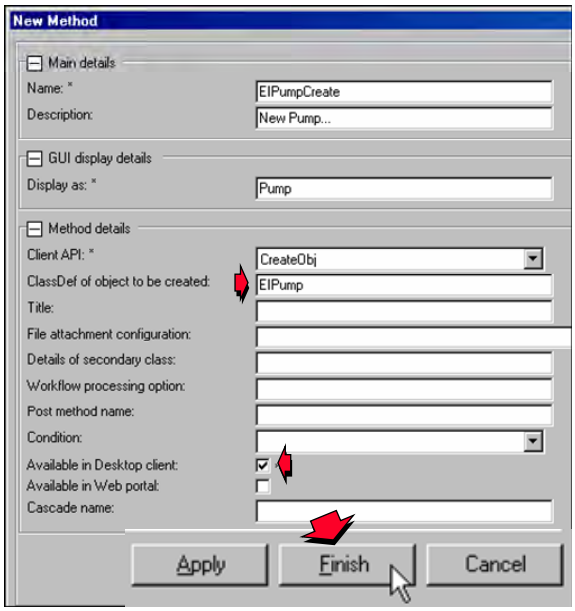
© 2008. Intergraph Corp.
All Rights Reserved.

Hover the cursor over an argument to see a long description of the API argument.

Creating Methods

- **Enter *EIPump* for the *ClassDef of object to be created.***

- **Enable the *Available in Desktop client* option and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

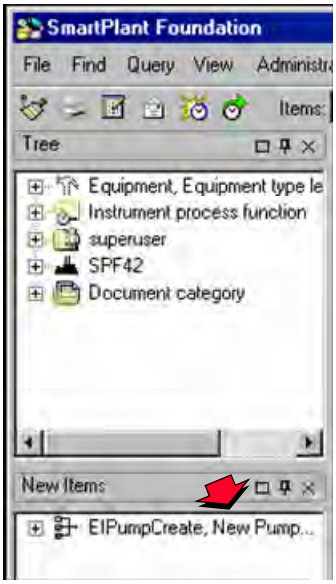
The *New method* form fields (con't):

- **Condition name** - optionally, defines a condition to run the API. If a condition is set on the method, the method does not appear on the shortcut menu for objects that do not meet the condition.
- **Available in Desktop client** - specifies that this method to be available in the Desktop Client interface.
- **Available in Web portal** - specifies that this method to be available in the Web portal interface.
- **Cascade name** - defines a submenu structure on the right click menu. Enter the name of the top-level menu in the field.

INTERGRAPH

Creating Methods

- **The new *EIPumpCreate* method displays in the *New Items* pane.**

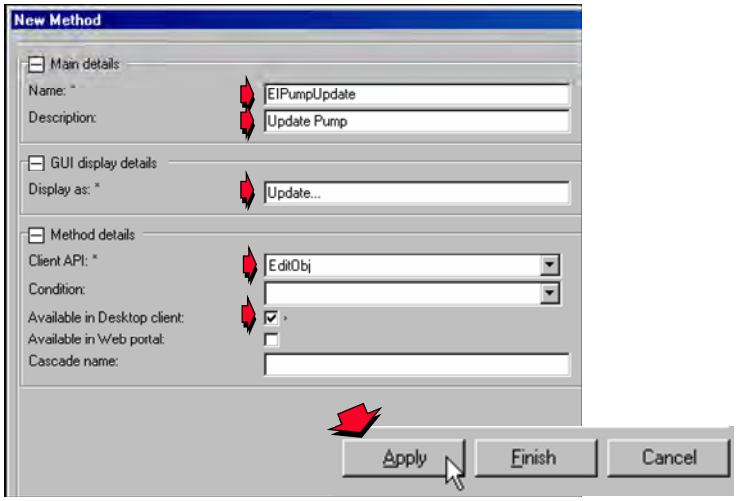


© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the steps to define an update method.

Creating Methods

- **Following the same procedure for *EIPumpCreate*, create another new method to update a pump object and click Apply.**

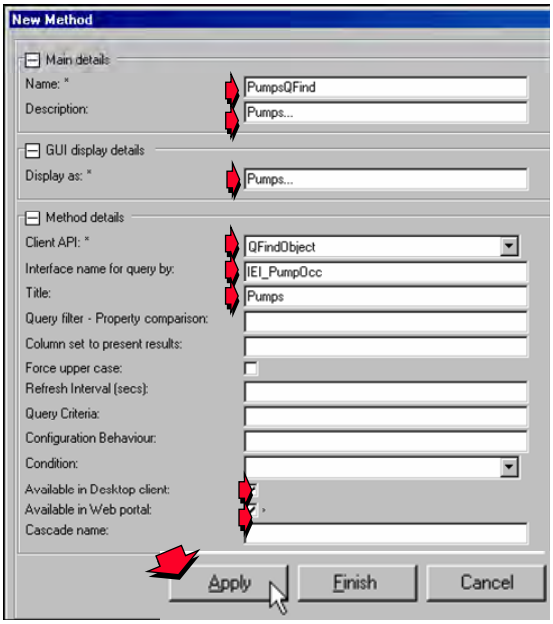


© 2008. Intergraph Corp.
All Rights Reserved.

Define a method that can be used to search for pump instances that have been created.

Creating Methods

- **Enter the necessary values to define the new method.**
- **Click the *Available in Web portal* option and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

The three new methods displayed can be used to create and manipulate pump tags.

Creating Methods

- **The new methods display in the *New Items* pane.**

© 2008. Intergraph Corp.
All Rights Reserved.

5.2.1 Common Method APIs

Below is a listing of some of the more common Client API's used with methods. Use the Desktop Client to find a complete list of Client API's along with a brief description and arguments for each API.

Common Menu Methods

Client API	Description	Arguments	
CreateObj	Used to Create Object	1	ClassDef of object to be created
LoadObject	Adds the named object to the tree view.	1	Class name
		2	Object name
		3	Query filter - Property comparison
		4	Column set to present results
QFindObject	Find an Object via its Interface.	1	Interface name for query by
		2	Title
		3	Query filter - Property comparison
		4	Column set to present results
QueryObject	Query Object.	1	ClassDef of object to be queried
		2	Refresh Interval

Common Shortcut Menu (RHM) Methods

Client API	Description	Arguments	
CopyObj	Copy object and relationships.	1	Title
		2	Copy files
		3	Show hidden files
		4	File attachment configuration
CreateObjAndRel	Used to Create Object and Relate it.	2	Relationship Definition to related object
DeleteObj	Used to delete the current highlighted object plus it's associated files.		
EditAllRels	Used to terminate all the relationships	1	Purpose = Terminate or Delete.
		2	Column set to all data.
		3	
EditObj	Used to edit an object	1	Update
			Sign Off (Revision Sign Off Form only).

Client API	Description	Arguments	
TerminateObj	Terminates the current highlighted object	1	IDEVAdminDoc ISPFDocumentMaster ISPFDocumentRevision ISPFDocumentVersion
ViewRelatedItems	Used to view related items. Shows main object.	1	No arguments

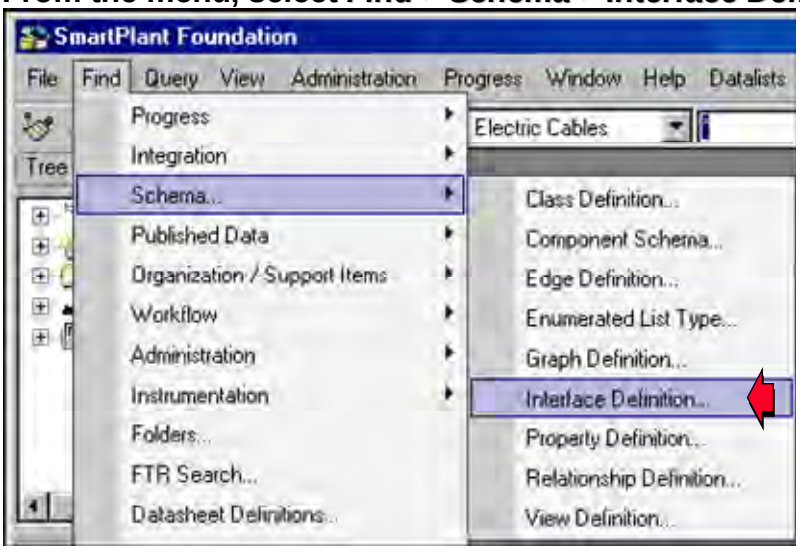
5.2.2 Manage Related Items

The *Manage Related Items* form provides the user with an interface to display and maintain related items for a specified relationship. Examples are *Manage Methods*, *Manage Related Items for Form*, *Manage Related Items for Column Sets*, and *Manage Menus*.

INTERGRAPH

Manage Related Items

- **From the menu, select *Find > Schema > Interface Definition.***



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Interface Definition* dialog appears.

Manage Related Items

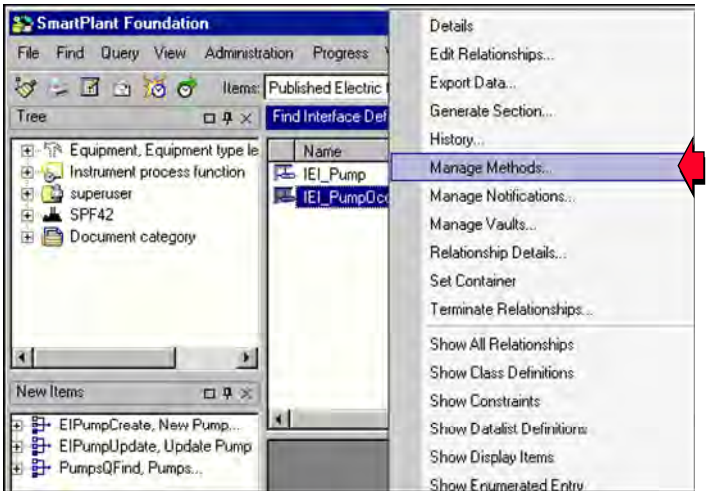
- **Enter *IEI_Pump** with the wildcard character (*) to find and display all interface definitions that start with *IEI_Pump*.**
- **Click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Use the *Manage Related Items* form to relate the new methods to an interface.

Manage Related Items

- **Right click on *IEI_PumpOccurences* and from the shortcut menu, select *Manage Methods*.**

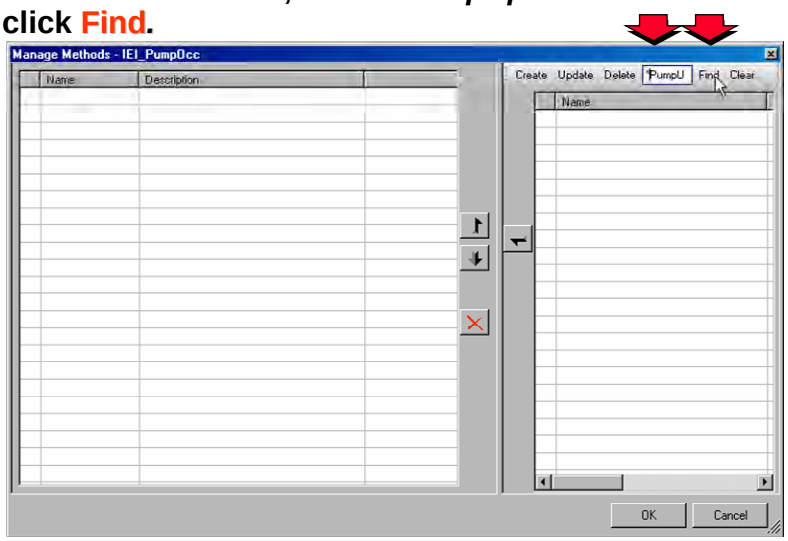


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Methods* version of the Manage Related Items form appears.

Manage Related Items

- **To find the method, enter **PumpUpd** in the search field and click Find.**



© 2008. Intergraph Corp.
All Rights Reserved.

The right pane displays a library that contains a set of valid items that can be related to the object. The library allows the user to *create* new items, *delete* items, and *find* and relate items to the original object.

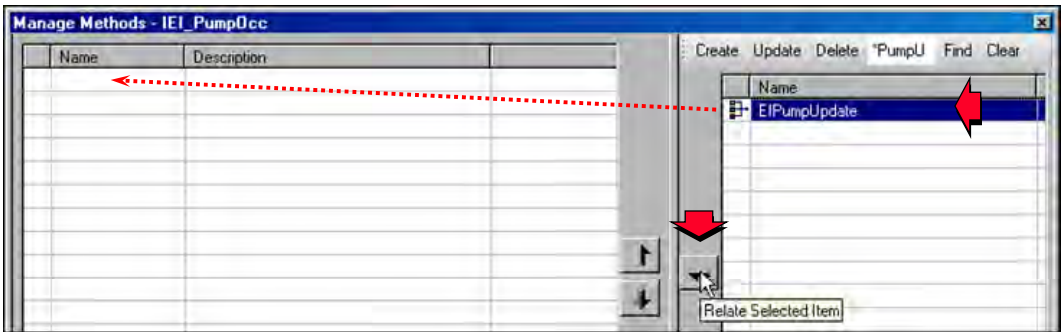
Library (Right) Pane Buttons

Create	**Create** - Opens a New Item form. On completion, the object is immediately saved to the database.
Update	**Update** - Opens the Update item form for the item selected in the library list. On completion, the object is immediately saved to the database.
Delete	**Delete**s the selected library item. On completion, the object is immediately removed from the database.
Find	**Find** - Enter an item name (or part name plus wildcards*) in the find field and click **Find**. The results are displayed in the library pane.
Clear	**Clear** - Clears the library pane.

The ***EIPumpUpdate*** method (created earlier) will appear on the right side of the form.

Manage Related Items

- **Select *EIPumpUpdate* and click the left arrow button to relate the item to the interface *IEI_PumpOcc*.**

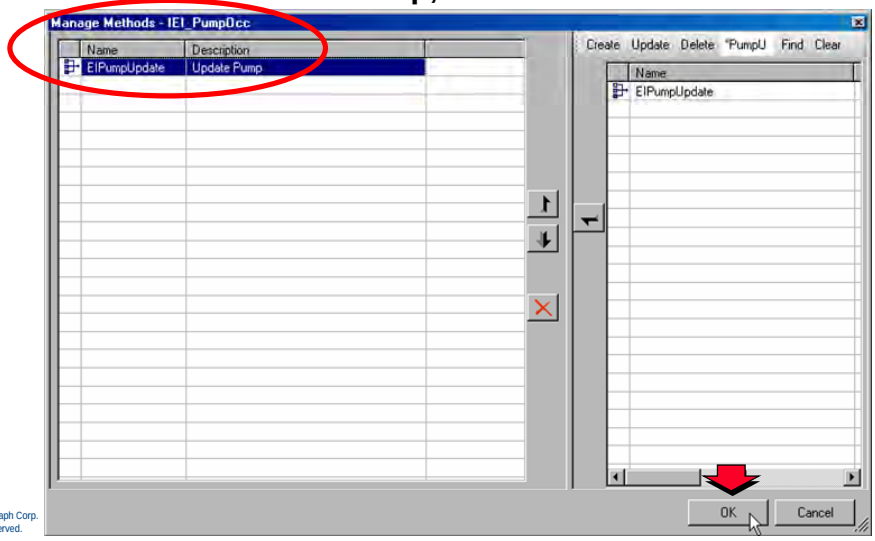


© 2008. Intergraph Corp.
All Rights Reserved.

The selected method will be displayed in the left pane of the form. The left pane represents a list of related items and the properties on the objects and the relationships that can be edited directly in the list view. These updates are saved to the database on selection of **OK** and ignored on selection of **Cancel**.

Manage Related Items

- **To save the relationship, click OK.**

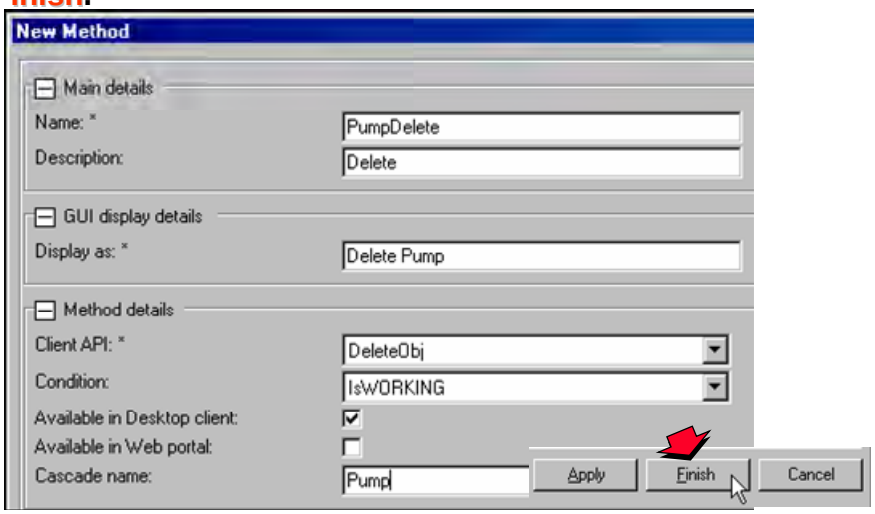


© 2008. Intergraph Corp.
All Rights Reserved.

Define another method to manipulate the pump class.

Creating Methods

- **Create a new method to delete a pump object and click Finish.**

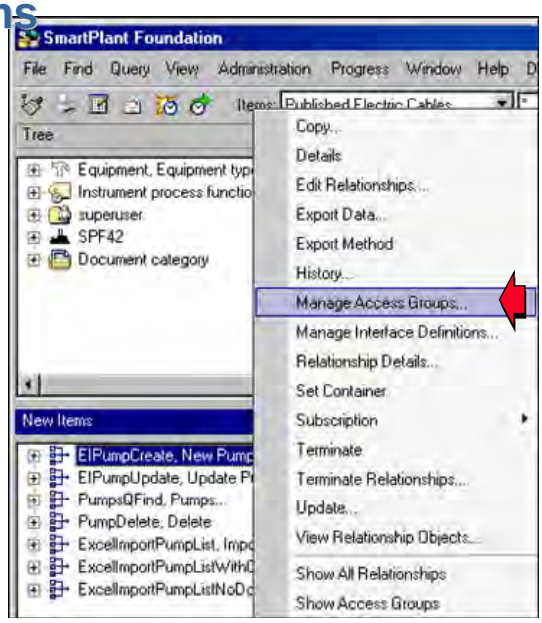


© 2008. Intergraph Corp.
All Rights Reserved.

Set up the method access control by relating the method to an **Access Group**. Use the *Manage Related Items* form to relate the new methods to access groups.

Manage Related Items

- **Right click on *ElPumpCreate* and from the shortcut menu, select *Manage Access Groups*.**

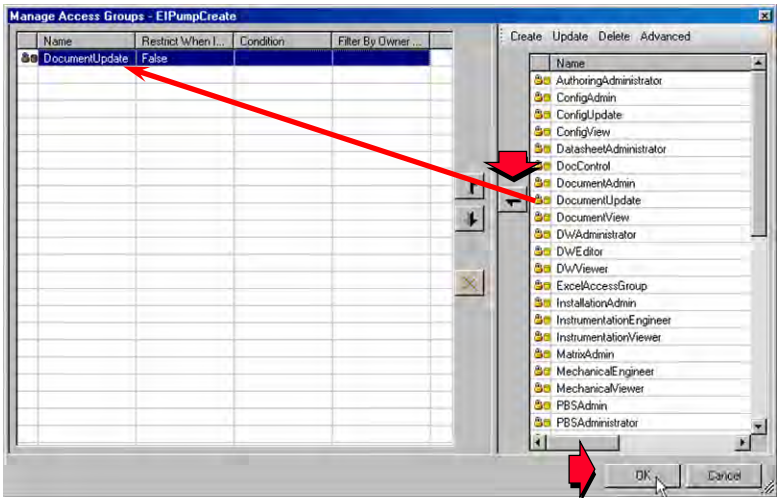


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Access Groups* version of the Manage Related Items form appears.

Manage Related Items

- **Select *DocumentUpdate* and click the left arrow button to relate this access group to *EIPumpCreate then* click OK.**

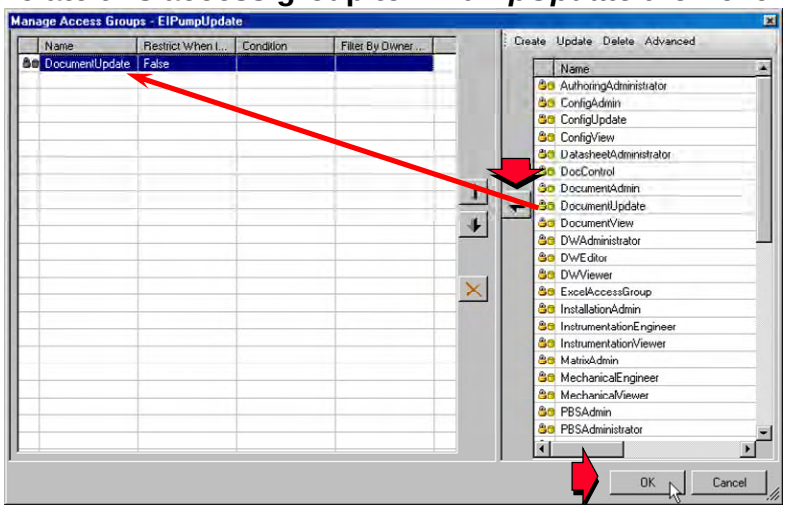


© 2008. Intergraph Corp.
All Rights Reserved.

Right click on *EIPumpUpdate* and from the shortcut menu, select ***Manage Access Groups***.

Manage Related Items

- **Select *DocumentUpdate* and click the left arrow button to relate this access group to *EIPumpUpdate then* click OK.**

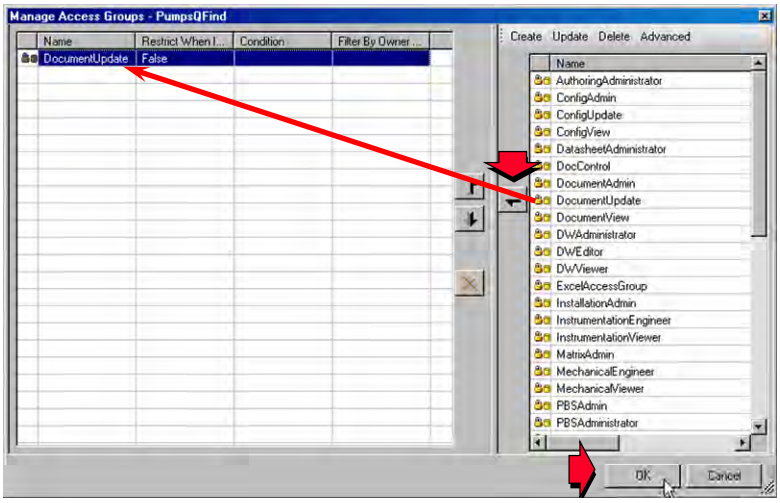


© 2008. Intergraph Corp.
All Rights Reserved.

Right click on *PumpsQFind* and from the shortcut menu, select ***Manage Access Groups.***

Manage Related Items

- **Select *DocumentUpdate* and click the left arrow button to relate this access group to *PumpsQFind then* click OK.**

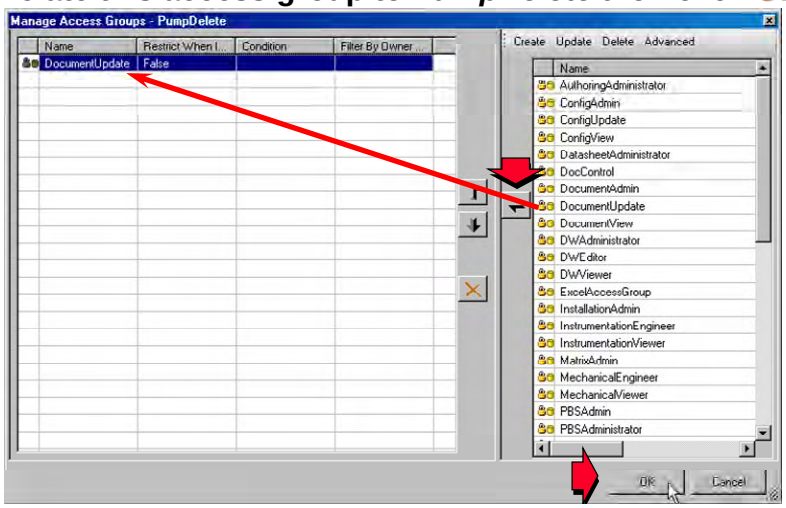


© 2008. Intergraph Corp.
All Rights Reserved.

Right click on *PumpDelete* and from the shortcut menu, select ***Manage Access Groups***.

Manage Related Items

- **Select *DocumentUpdate* and click the left arrow button to relate this access group to *PumpDelete then* click OK.**



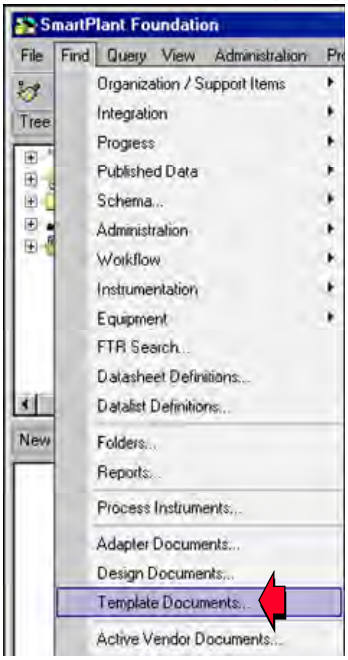
© 2008. Intergraph Corp.
All Rights Reserved.

5.2.3 View Related Items

The *View Related Items* functionality displays the details form for the selected object and one tab for each relationship the user can navigate. Start by finding an object.

View Related Items

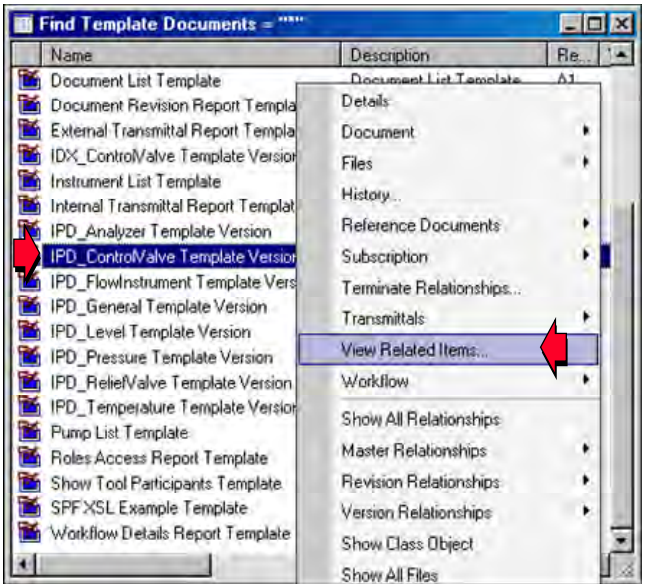
- **Select *Find > Template Documents* from the menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Template Documents* list view appears.

View Related Items

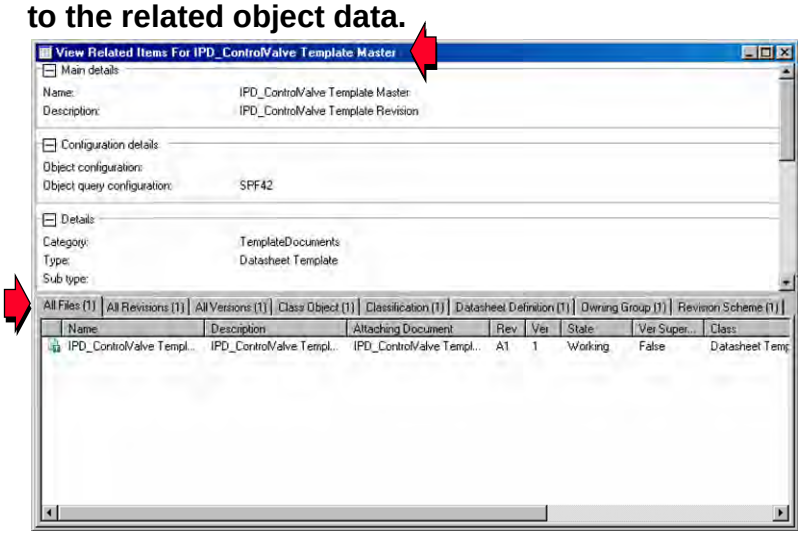


- **Right click on a template document (IPD_ControlValve Template Version) and from the shortcut menu, select *View Related Items.***

The main object will display in the details form with tabs for each related object type below.

View Related Items

- **Select a relationship tab from the details form to navigate to the related object data.**

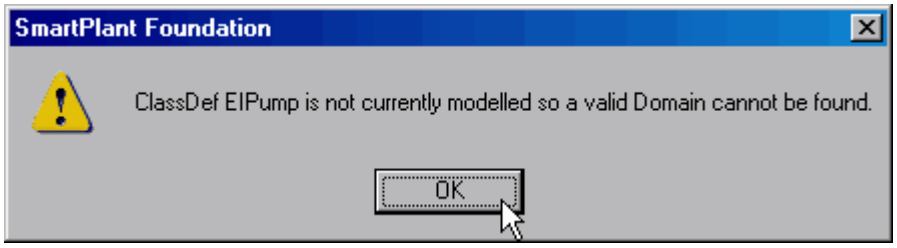


© 2008. Intergraph Corp.
All Rights Reserved.

5.2.4 Relating New Classes to a Domain

Before a new class can be used with menus and forms, it must be related to a defined domain. This is done via a relationship with the class's component schema object and a domain object.

Failure to relate a class's component schema to a domain will result in an error when methods are executed on the class. The following is the message that is displayed.

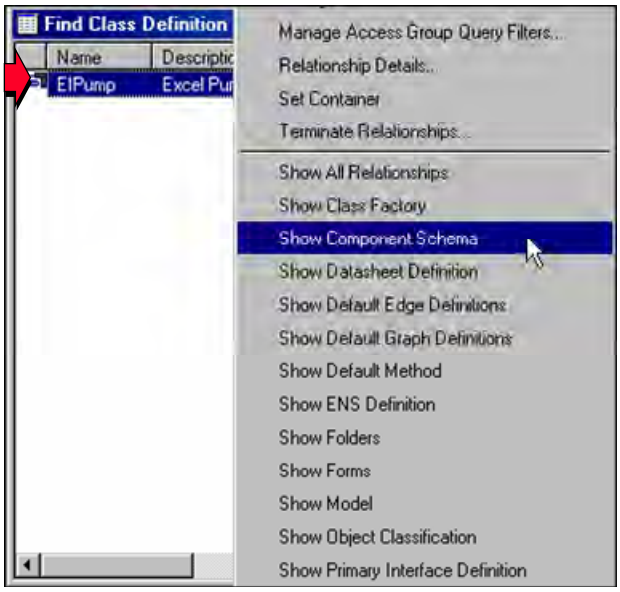


The first step is to view the component schema for the pump class.

INTERGRAPH

Relating New Classes to a Domain

- **Right click on the *ElPump* ClassDef and select *Show Component Schema* from the right mouse menu.**

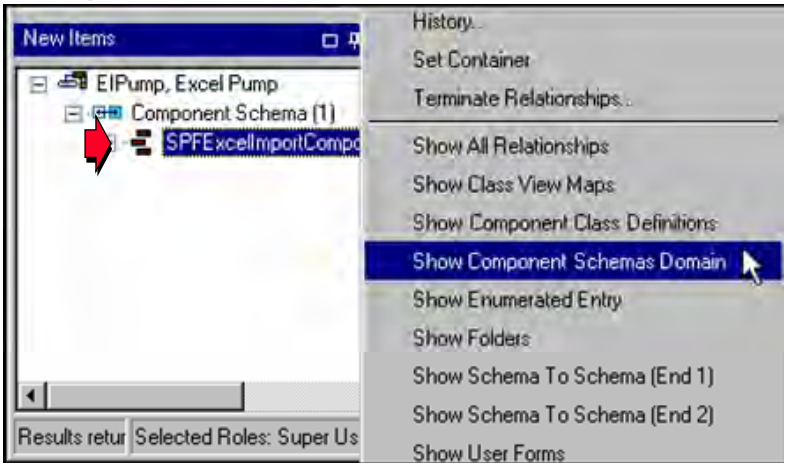


© 2008. Intergraph Corp.
All Rights Reserved.

The tree will expand to show the related component schema.

Relating New Classes to a Domain

- **Right click on the *SPFExcelImportComponent* Component Schema and from the right mouse menu, select *Show Component Schemas Domain*.**

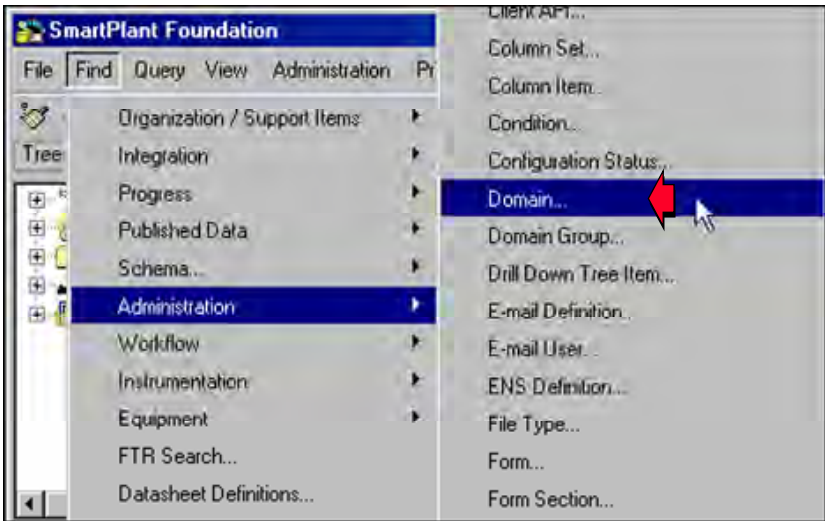


© 2008. Intergraph Corp.
All Rights Reserved.

Note that nothing will display since the component schema does not have a relationship to the domain. Use the menu to perform a search for an existing domain.

Relating New Classes to a Domain

- **Select *Find > Administration > Domain* from the menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Domain* dialog will appear.

Relating New Classes to a Domain

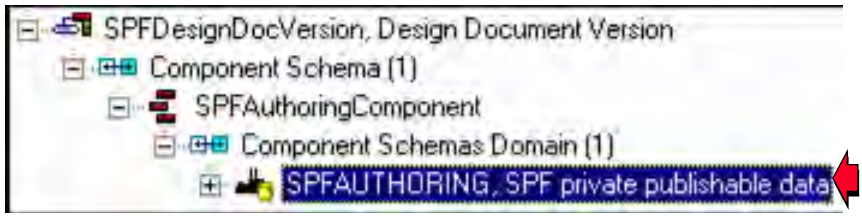
- **Enter *SPF** to find and display some of the existing domains and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

In order to relate the component schema to the correct domain, use an existing class, such as *Design Document*, as an example.

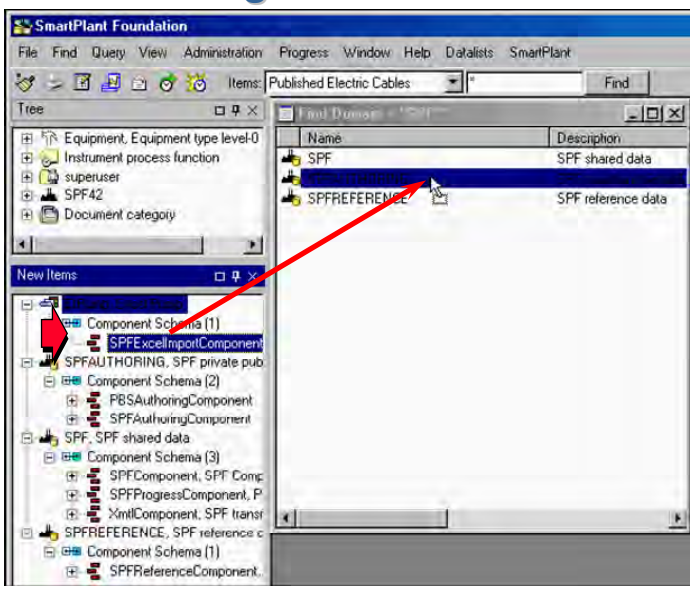
Relating New Classes to a Domain

- **Locate and review an existing class to determine which domain to use for the new class.**



© 2008. Intergraph Corp.
All Rights Reserved.

Relating New Classes to a Domain

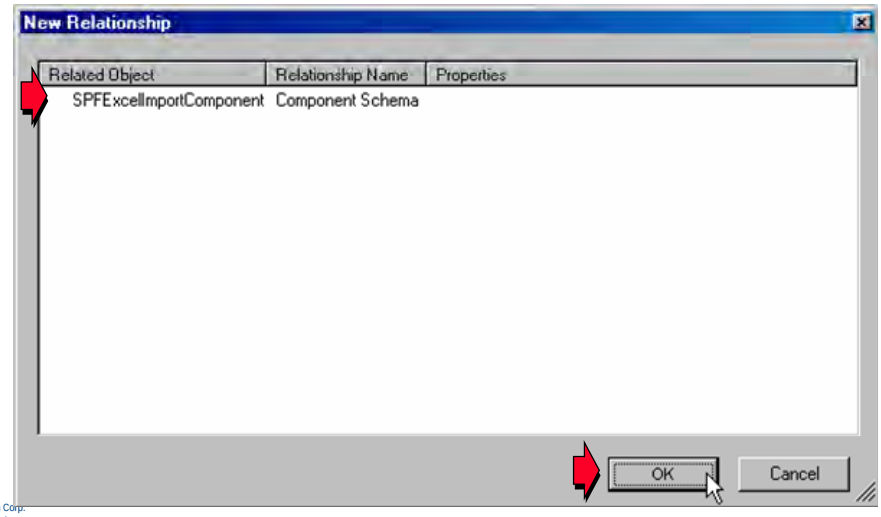


- **Drag and drop the *SPFExcelImport Component* component schema onto the *SPFAUTHORING* domain.**

The *New Relationship* form will appear.

Relating New Classes to a Domain

- **Verify that the new relationship is correct and click OK.**

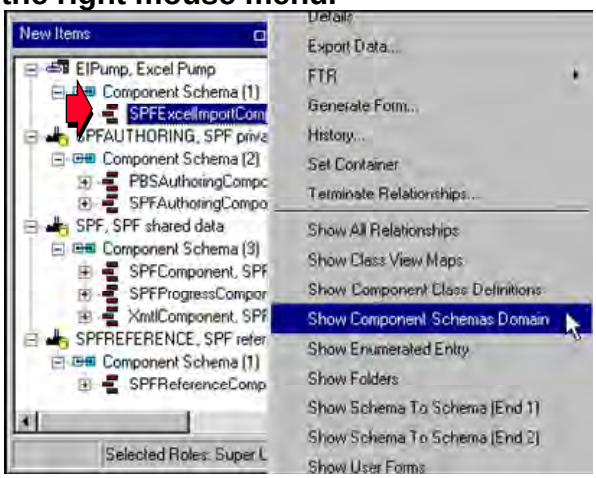


© 2008. Intergraph Corp.
All Rights Reserved.

To view the new relationship, expand the *SPFExcelImportComponent* component schema object to view the related domain.

Relating New Classes to a Domain

- **Right click on the *SPFExcelImportComponent* component schema and select *Show Component Schemas Domain* from the right mouse menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

The tree should now expand to show the related domain.

Relating New Classes to a Domain

- **Verify that the *SPFExcelImport Component* Component Schema is now related to the *SPFAUTHORING* domain.**

5.3 Activity 1 – Configuring New Methods

Complete **Chapter 5 - Activity 1** in the SPF Configuration and Administration II activity workbook.

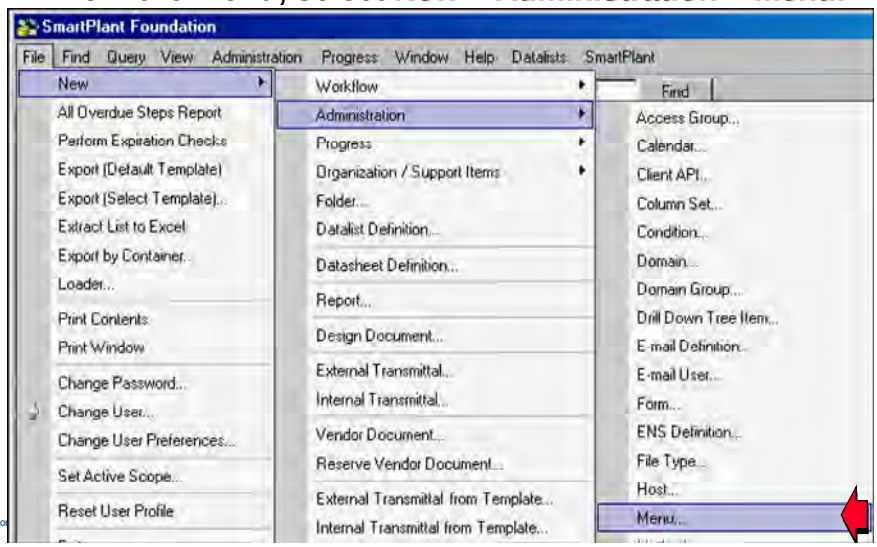
5.4 Creating and Managing Menus and Menu Items

Menus are used to group various actions performed in the SmartPlant Foundation Client. Menu items (actions) can be created for each menu node specifying the action to be performed, the method to be used, and other related properties. In SmartPlant Foundation, menus are static and are always displayed, regardless of the user group.

To create a new menu:

Creating Menus

- **From the menu, select *New > Administration > Menu*.**

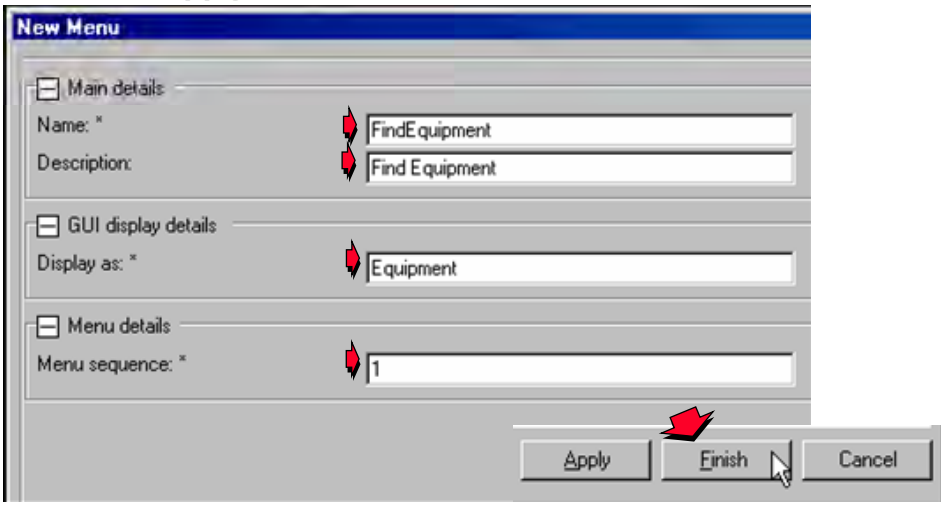


© 2008. Intergraph Co
All Rights Reserved.

The *New Menu* form is displayed.

Creating Menus

- **Enter the values for the new menu, set *Menu sequence* to 1 and click Apply.**



© 2008. Intergraph Corp.
All Rights Reserved.

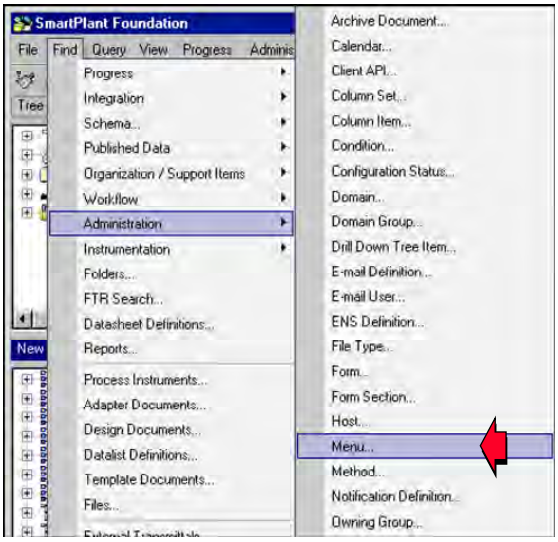
The *New Menu* form contains the following fields:

- **Name** - specifies the name/purpose of the menu and is a required field.
- **Description** - specifies a description for the menu and is an optional field.
- **Display as** - specifies the name that will be displayed on the SmartPlant Foundation Desktop Client menu bar. This is a required field.
- **Menu sequence** - defines a sequence number for the menu node. Sequence numbers define the position of the menu node on the menu going from left to right.

Perform a search for the *New* menu node in order to add the ability to create a new pump object.

Managing Menu Items

- **From the menu, click *Find > Administration > Menu*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Menu* dialog will appear.

Managing Menu Items

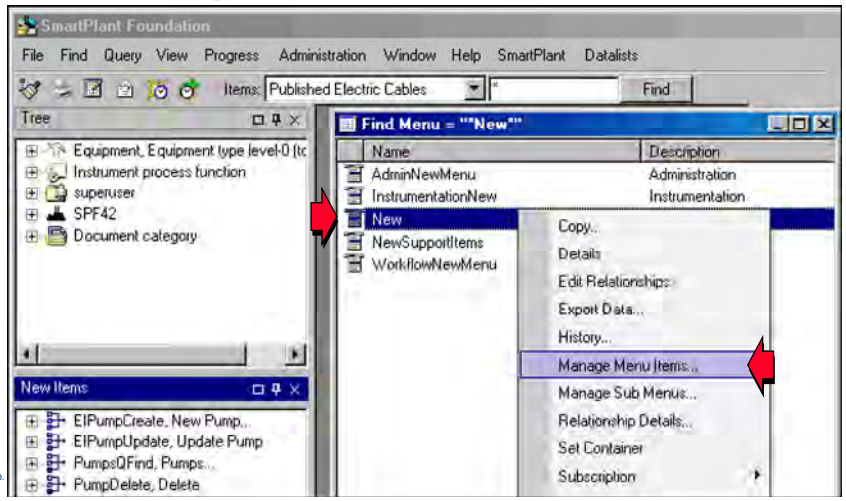
- **Enter *New* with the wildcard character, *, to find and display all Menus that contain *New* and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Locate the *New* menu node in the list view.

Managing Menu Items

- **Right click on the *New* menu and from the shortcut menu, select *Manage Menu Items*.**

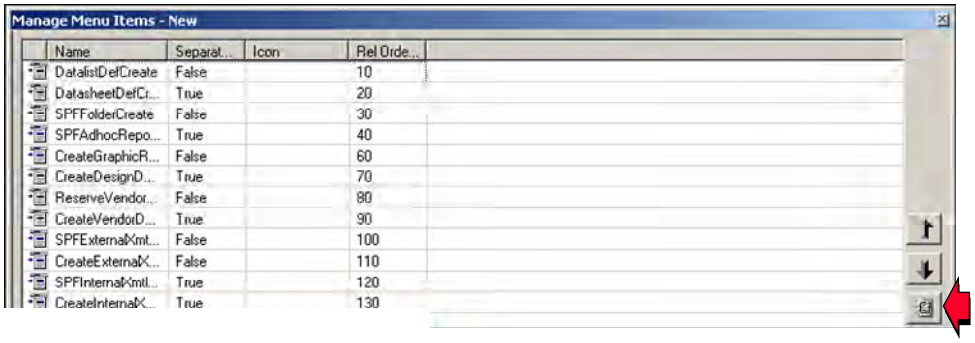


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Menu Items – New* dialog appears.

Creating Menu Items

- **On the *Manage Menu Items* dialog, click the create button.**

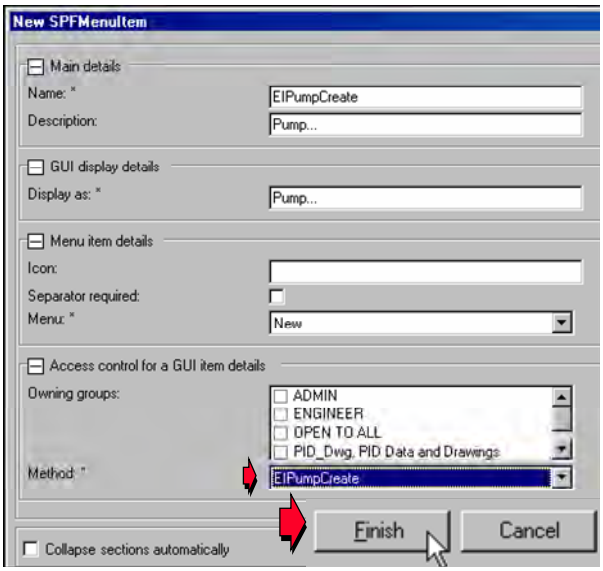


© 2008. Intergraph Corp.
All Rights Reserved.

Enter the required values to define a menu item to create a new pump.

Managing Menu Items

- **Select *EIPumpCreate* as the *Method* and click Finish.**

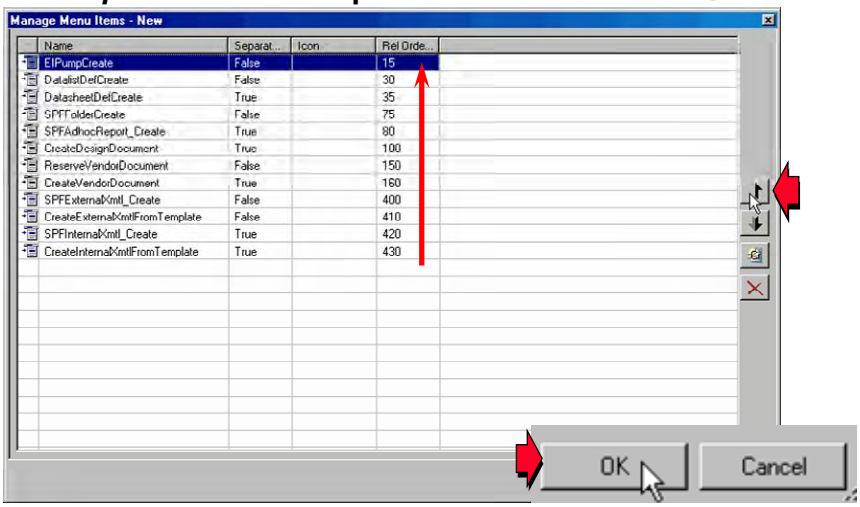


© 2008. Intergraph Corp.
All Rights Reserved.

To change the sequence of an item in the menu, select the item and use the *up* and *down* arrow buttons on the form to move the item.

Managing Menu Items

- **Select the *Menu Item* and click the up arrow button to move *EIPumpCreate* to the top of the menu the click OK.**

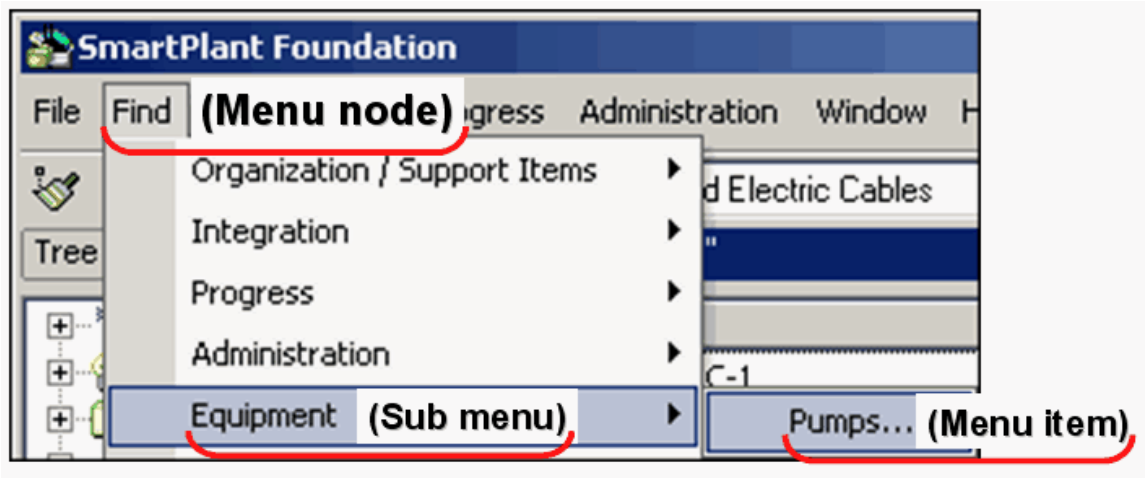


© 2008. Intergraph Corp.
All Rights Reserved.

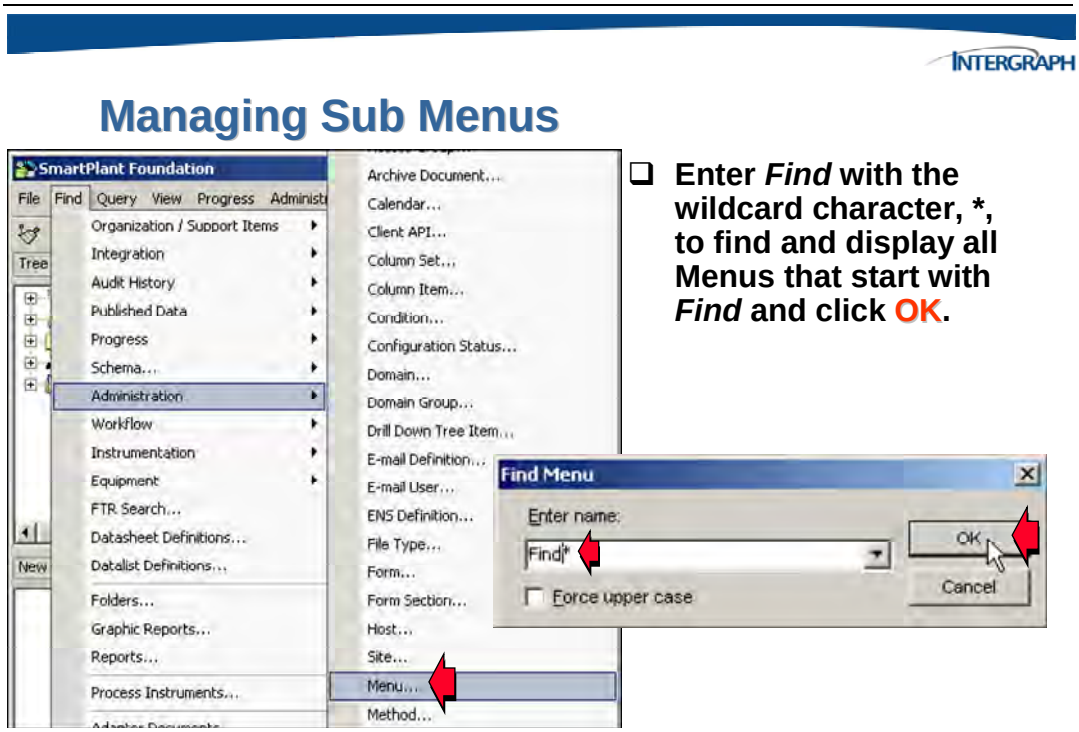
5.4.1 Managing Sub Menus

Sub menus are created by relating one menu (sub menu) to another (parent menu node).

The new sub menu structure will look like this:



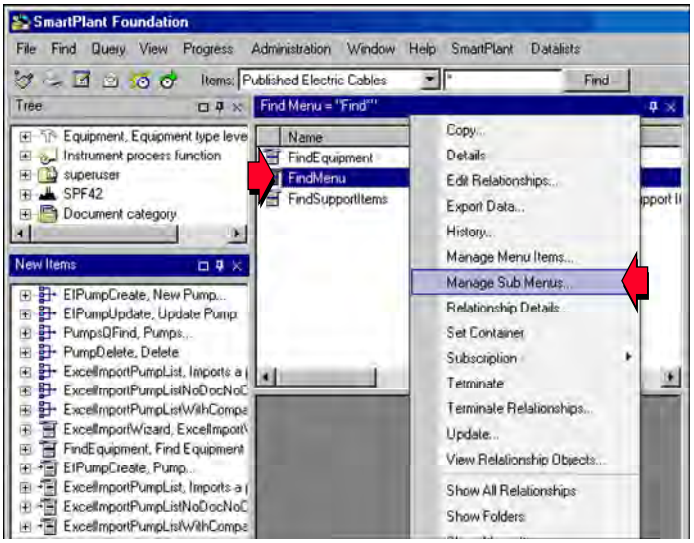
Perform a search for the *Find* menu node in order to create a set of sub menus, very similar to the *New* sub menu.



The existing *FindMenu* menu node will display in the list view browser.

Managing Sub Menus

- **Right click on the *FindMenu* and from the shortcut menu, select *Manage Sub Menus.***

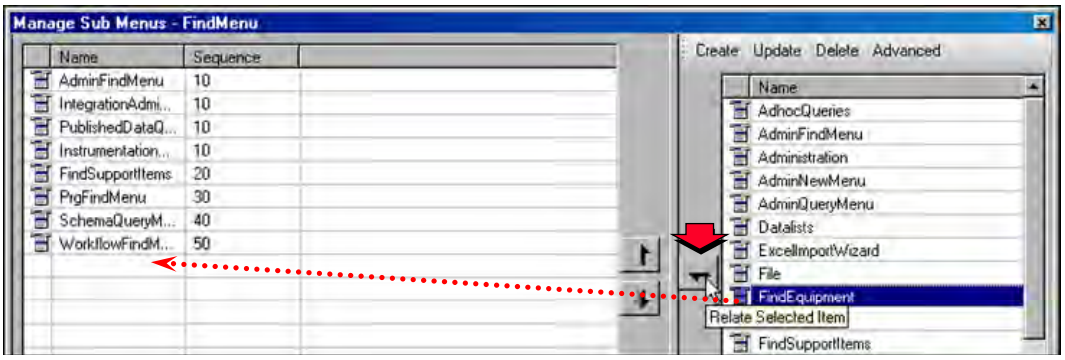


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Sub Menus* form can be used to manage the sub menu (that is create, delete, and change the sequence of the sub menu). The *Manage Sub Menus* form will appear.

Managing Sub Menus

- **Select *FindEquipment* and click the left arrow button to relate it to the submenu *FindMenu*.**

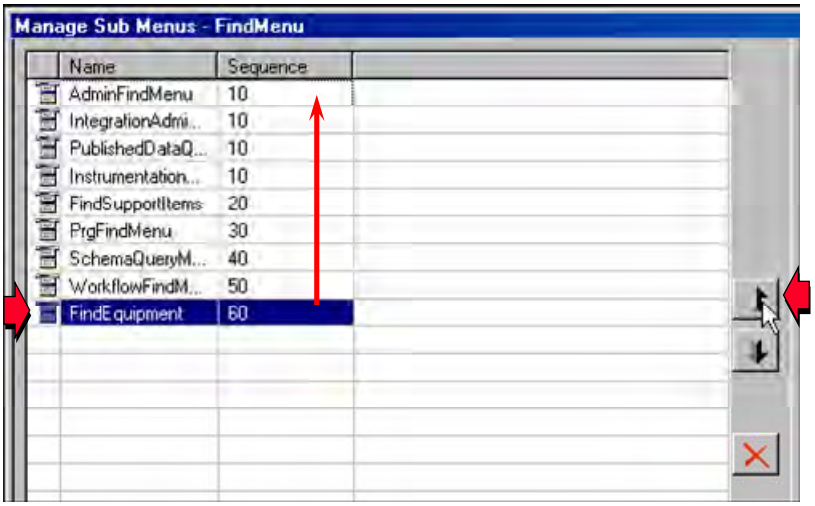


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *up* and *down* arrows on the form to change the sub menu sequence.

Managing Sub Menus

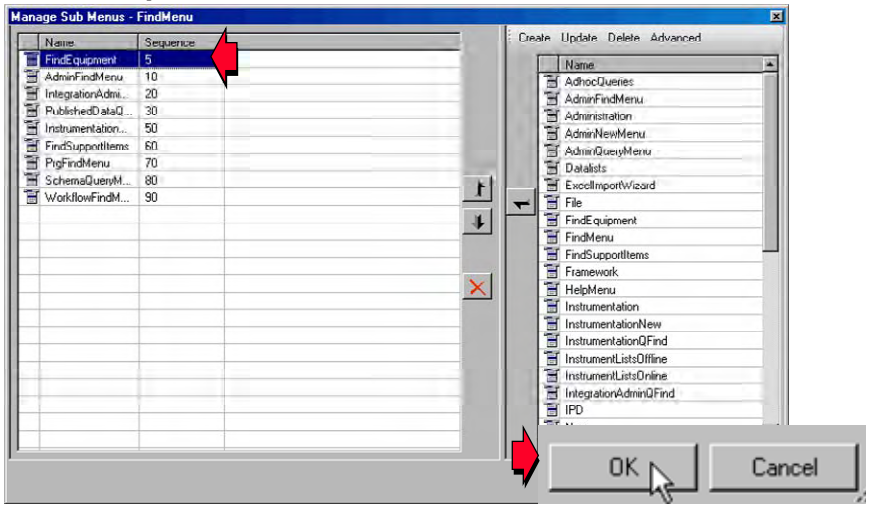
- **Use the up arrow to resequence the *FindEquipment* menu item.**



© 2008. Intergraph Corp.
All Rights Reserved.

Managing Sub Menus

- **After *FindEquipment* has been moved to the top of the menu sequence, click OK.**

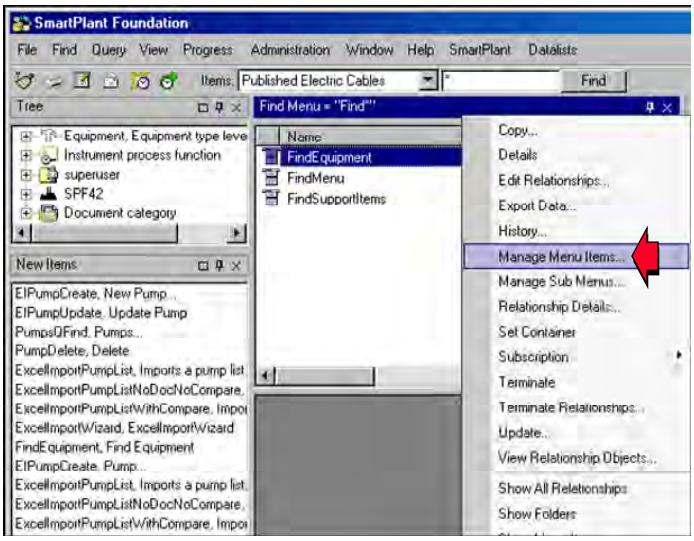


© 2008. Intergraph Corp.
All Rights Reserved.

Create a new menu item in order to launch the **PumpsQFind** method.

Managing Menu Items

- **Right click on *FindEquipment* and from the shortcut menu, select *Manage Menu Items*.**

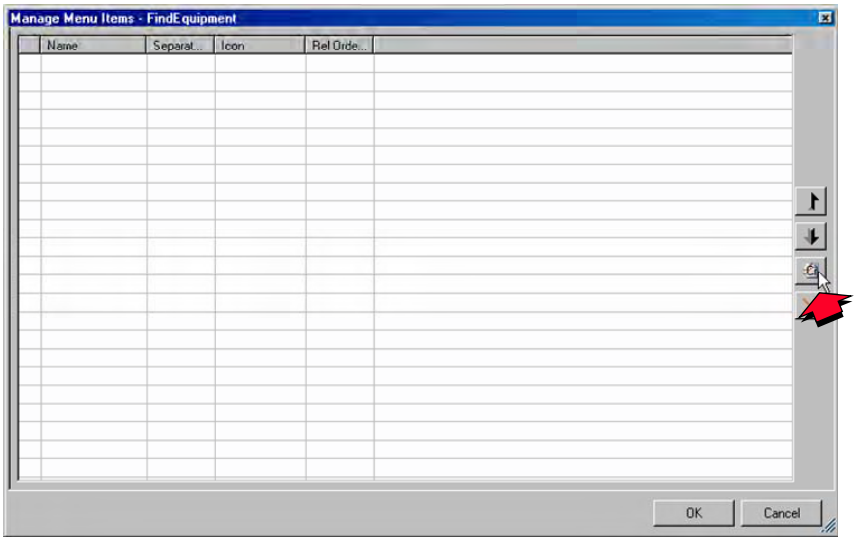


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Menu Items* form will appear.

Creating Menu Items

- **On the *Manage Menu Items* form, click the *Create* button.**

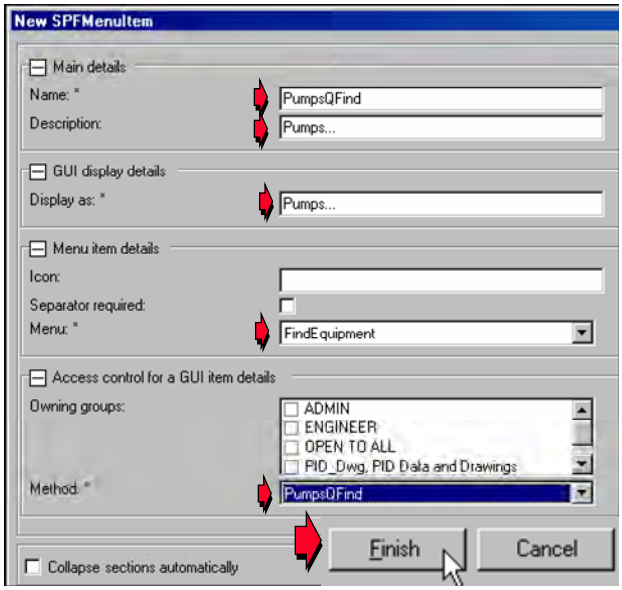


© 2008. Intergraph Corp.
All Rights Reserved.

The *New SPFMenuItem* form will appear.

Creating Menu Items

- **Enter appropriate values to define a new *Menu Item* and click Finish.**

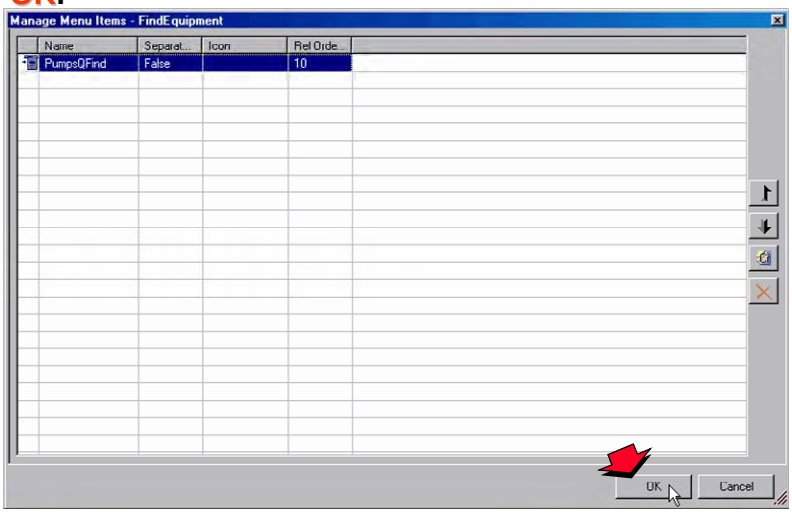


© 2008. Intergraph Corp.
All Rights Reserved.

This will add the **PumpsQFind** menu item to the *FindEquipment* menu node.

Creating Menu Items

- **Verify the *PumpQFind* menu item is created and click OK.**

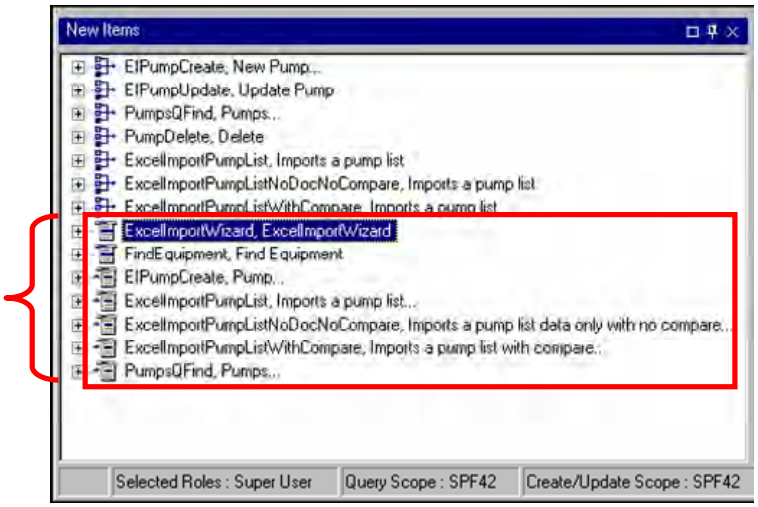


© 2008. Intergraph Corp.
All Rights Reserved.

The stand alone menu commands to create and find pump objects have now been configured.

Creating Menu Items

The new menus and menu items will be displayed in the *New Items* window.



© 2008. Intergraph Corp.
All Rights Reserved.

Note: Even though three methods for *ExcelImport* are shown above, configuring these methods will be discussed later in this chapter.

5.5 Creating Toolbars

Toolbar items provide a shortcut method for performing various actions in an SPF Client. Before the toolbar is created, a method must be defined for use with the toolbar item.

When a toolbar item is created, identify:

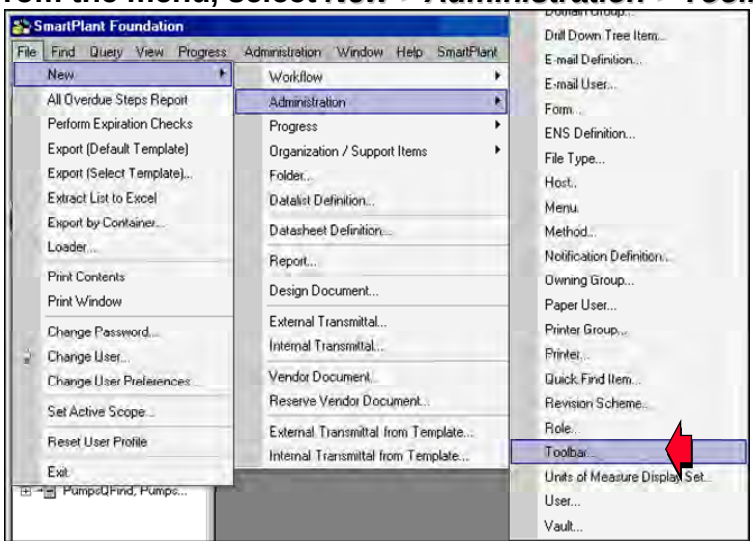
- ❑ The method to be used.
- ❑ The tool tip text.
- ❑ The icon to be displayed on the toolbar (if any).
- ❑ The item's location (sequence) on the toolbar.

To create a new toolbar item:

INTERGRAPH

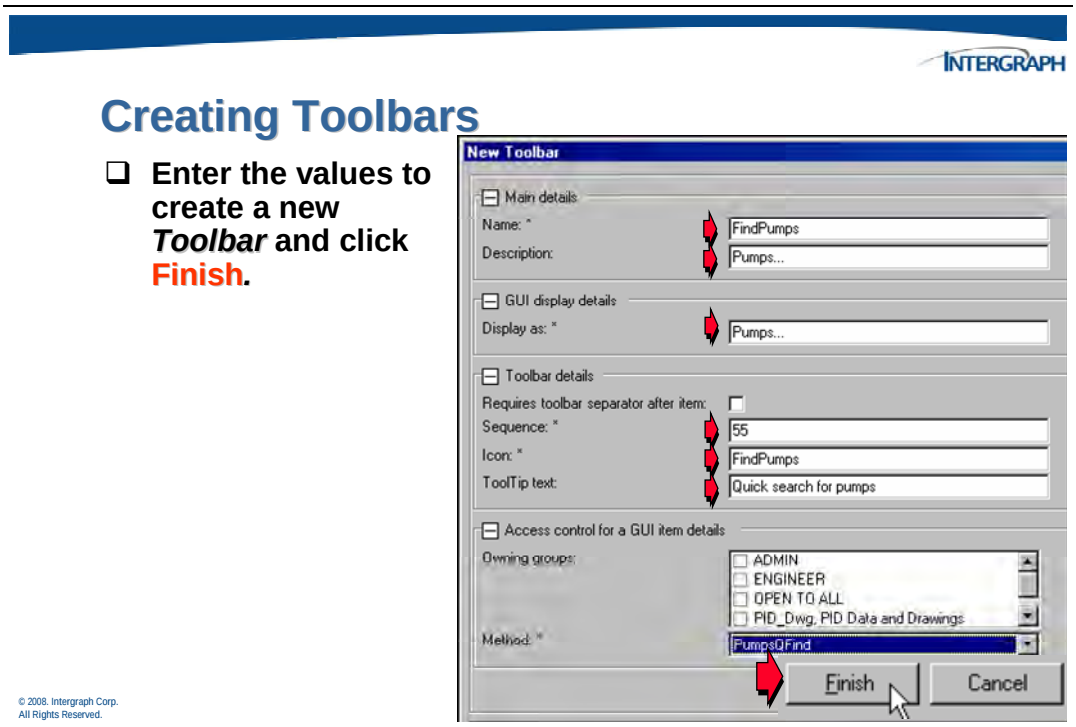
Creating Toolbars

❑ **From the menu, select *New > Administration > Toolbar*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The **New Toolbar** form will appear.



The *New Toolbar* form contains the following fields:

- **Name** - specifies the name/purpose of the toolbar and is a required field.
- **Description** - specifies a description for the toolbar and is an optional field.
- **Display as** - specifies the name that will be displayed on the SmartPlant Foundation Desktop Client toolbar. This is a required field.
- **Requires toolbar separator after item** - select the check box to insert a separator between this toolbar item and other toolbar items after it.
- **Sequence** - enter a value to determine the order of appearance on the toolbar. This is a required field.
- **Icon** - associates an icon with the toolbar item. Icons must be located in the \SmartPlant Foundation 2008 Server Files\Web_Sites\<site name>\Icons folder. Icons must be in .gif format and is a required field.
- **ToolTip text** - enter the help/informational text to be displayed.
- **Owning groups** - select the owning group that can access this toolbar. Selecting an owning group will restrict who can access the toolbar. This is not to be confused with access group control that is configured on the related method. This is an optional field.
- **Method** - select the method to be called from the pull down menu. The method must have an access group and is a required field.

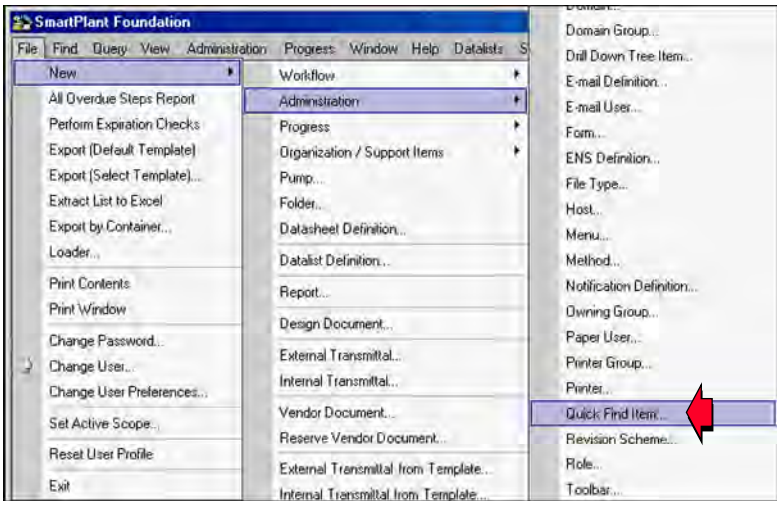
5.6 Quick Find Menu Items

The *Quick Find* menu lists all the Quick find methods that a user has access to through their role permissions.

To add a new item to the Quick Find menu.

Quick Find Menu Items

- **From the menu, click *File > New > Administration > Quick Find Item*.**

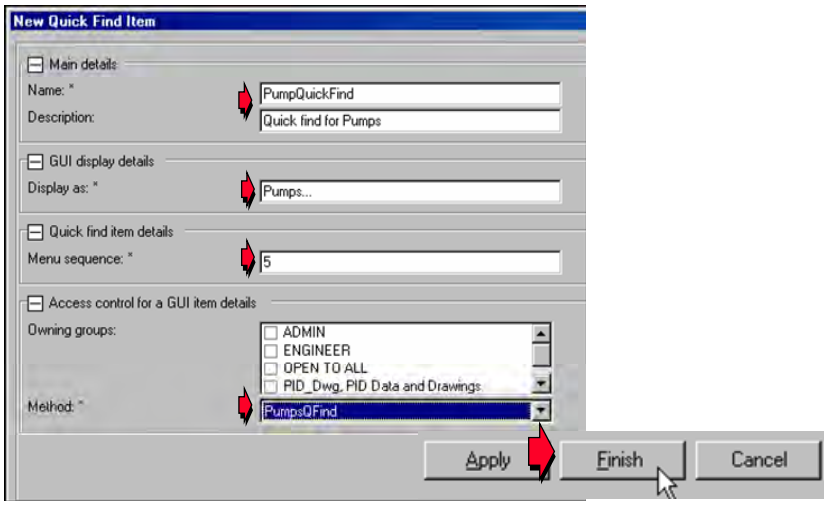


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Quick Find Item* form will appear.

Quick Find Menu Items

- **Enter the appropriate values to define the new *Quick Find* menu item then click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

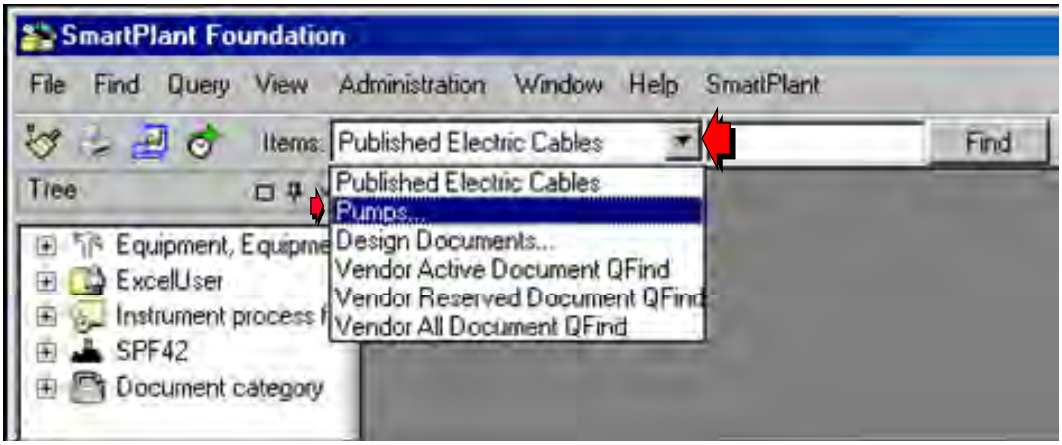
The *New Quick Find Item* form contains the following fields:

- **Name** - specifies the name/purpose of the item and is a required field.
- **Description** - specifies a description for the item and is an optional field.
- **Display as** - specifies the text that will be displayed on the Quick Find menu. This is a required field.
- **Menu sequence** - specifies the sequence of this item on the Quick Find menu and is a required field.
- **Owning groups** - select the owning group that can access this item. Selecting an owning group will restrict who can access the item. This is an optional field.
- **Method** - choose the method to be run when this item is selected on the Quick Find menu. The method must have an access group and is a required field.

Log in as a different user to see the results of the *Quick Find Item*.

Quick Find Menu Items

- **Open the Items pick list from the toolbar to see the new *Pumps Quick Find* menu item.**



© 2008. Intergraph Corp.
All Rights Reserved.

5.7 Activity 2 – Creating Menus and Menu Items

Complete **Chapter 5** - **Activity 2** in the SPF Configuration and Administration II activity workbook.

5.8 Tree View Display Column Set

The tree view by default displays *Name* and *Description* based on a user preference. Sometimes there is a need for additional properties; this is configured by creating a **column set** and relating it to the class Definition to be displayed. An example is for documents.

SPFDocumentVersionTreeDisplay is related to the object's Class Definition by the SPFClassDefDisplayColumnSet Relationship Definition.

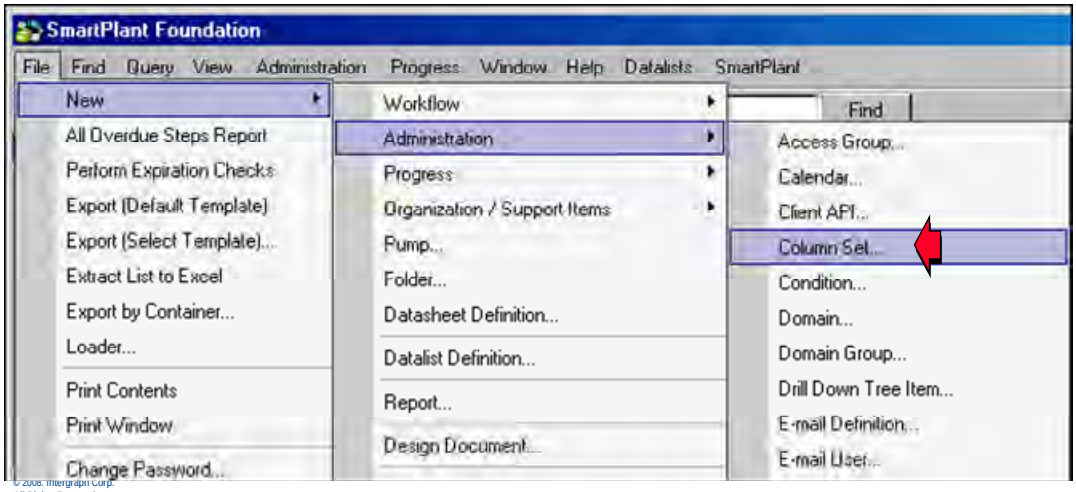
- ❑ Name
- ❑ Revision
- ❑ Version
- ❑ Workflow Status

To create a new column set:

INTERGRAPH

Creating Column Sets

- ❑ **From the menu, click *File > New > Administration > Column Set*.**

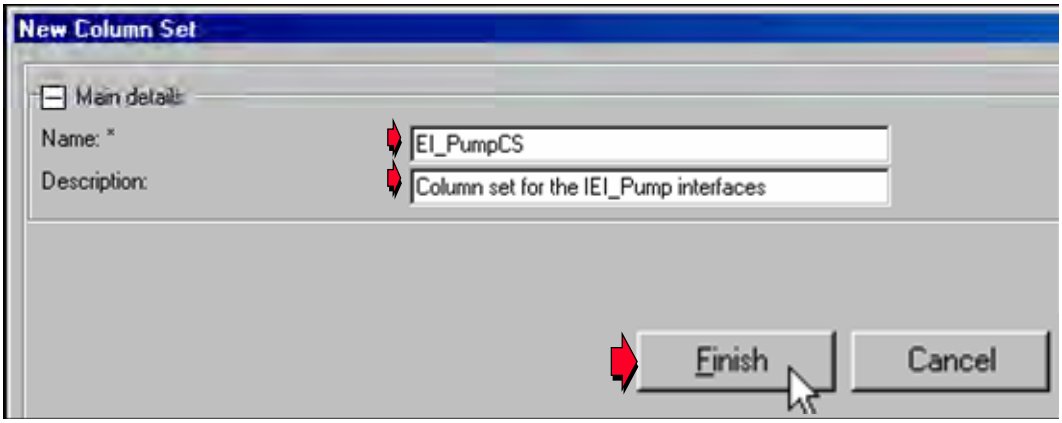


© 2008 Intergraph Corp.
All Rights Reserved.

The *New Column Set* form will appear.

Creating Column Sets

- **Enter the appropriate values to define a new *Column Set* and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

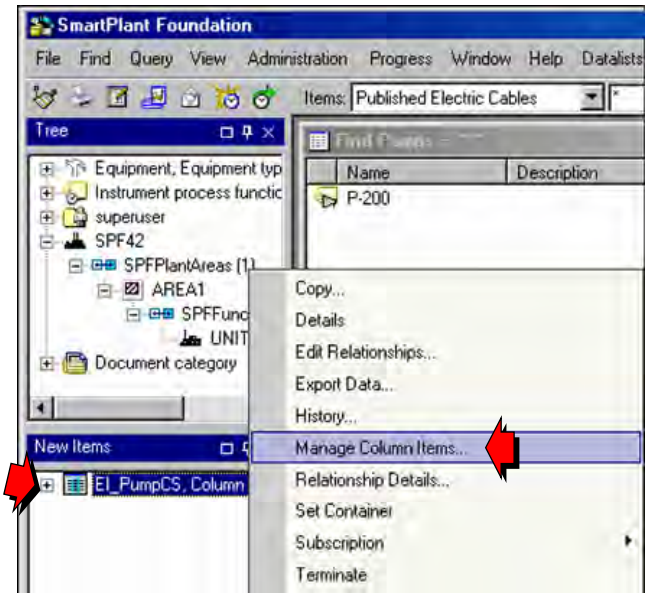
The *New Column Set* form contains the following fields:

- **Name** - specifies the name/purpose of the column set and is a required field.
- **Description** - specifies a description for the column set and is an optional field.

Once the column set has been created, column items can be added to it.

Manage Column Items

- **Right click on the new column set and select *Manage Column Items* from the shortcut menu.**

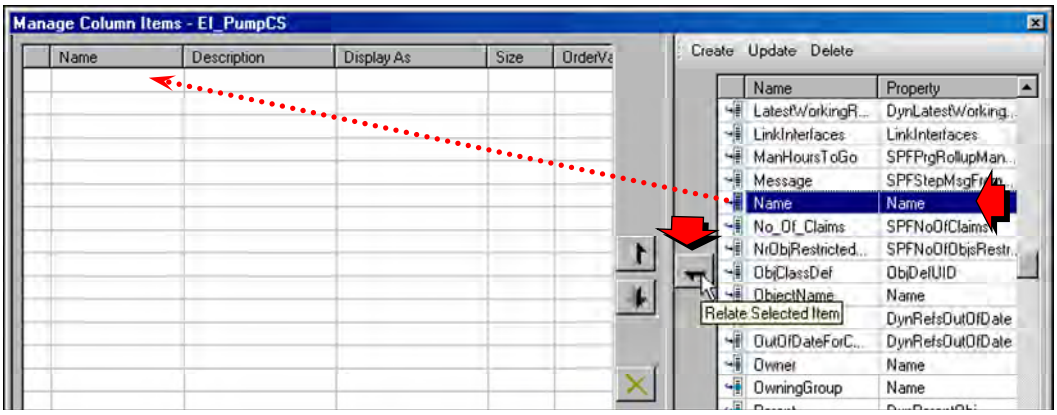


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Column Items* form will appear.

Manage Column Items

- **Select the *Name* column item and click the left arrow button to relate the item to the column set.**



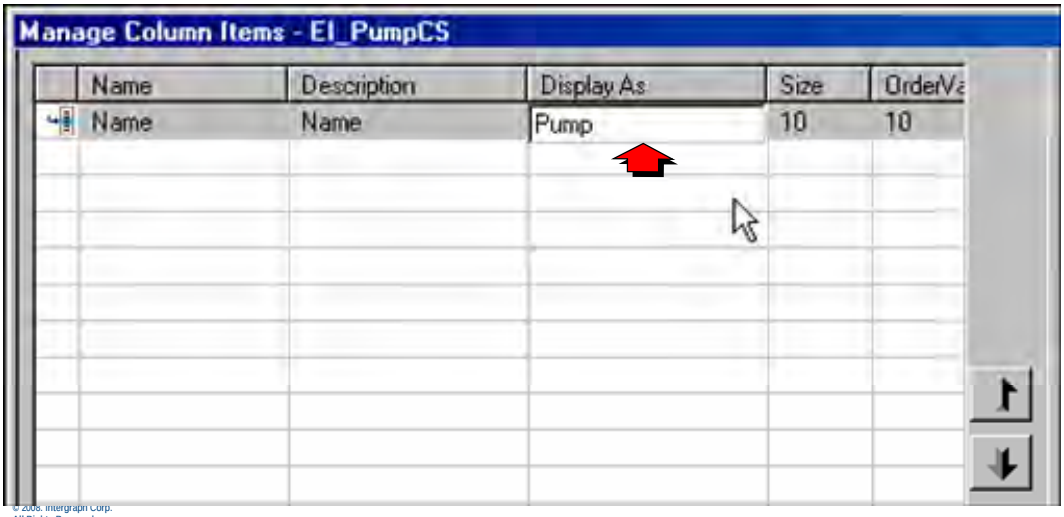
© 2008. Intergraph Corp.
All Rights Reserved.

The **Name** is always needed. The rest of the properties are displayed in the *Library* pane.

The *DisplayAs*, *Size*, and *OrderValue* can be edited directly by clicking in the box.

Manage Column Items

- **Change the *Display As* field to show the label as a Pump.**

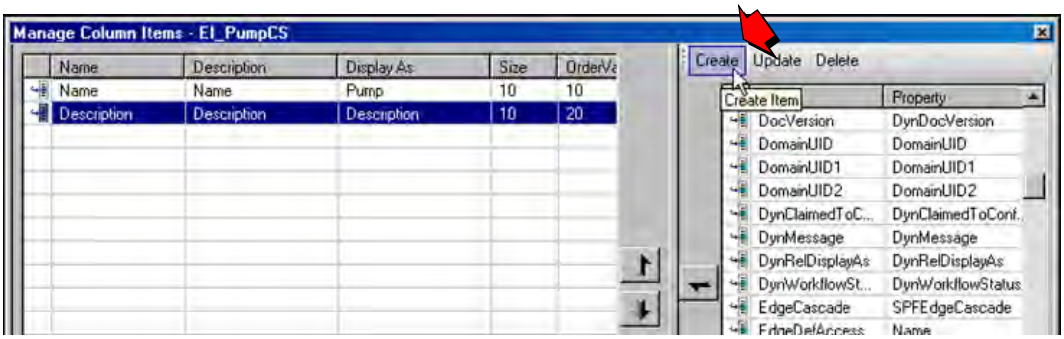


© 2008. Intergraph Corp.
All Rights Reserved.

Next, create new column items that can be used to display the custom pump properties.

Manage Column Items

- **To create a new column item, click *Create*.**

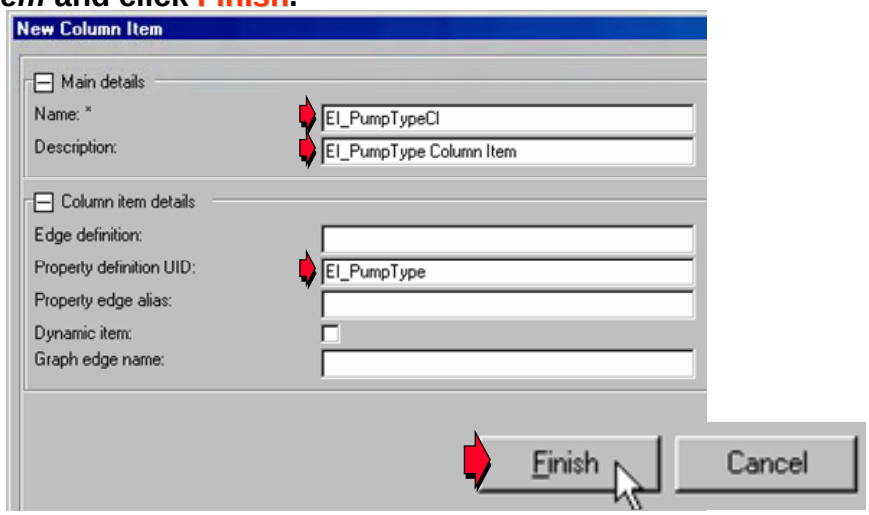


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Column Item* form will appear.

Manage Column Items

- **Enter the appropriate values to define the new *Column Item* and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

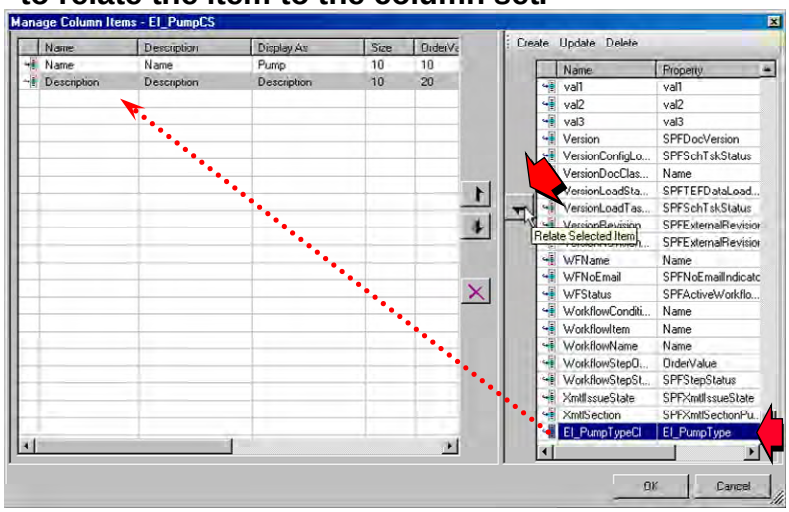
The *New Column Item* form contains the following fields:

- **Name** - Specifies the name/purpose of the column item and is a required field.
- **Description** - Specifies a description for the column item and is an optional field.
- **Edge definition** - The name of the edge definition that the column item will display.
- **Property definition UID** - The UID of the property definition that the column item will display, for example, the property name.
- **Property edge alias** - The alias used for the relationship between the property definition and the edge definition.
- **Dynamic item** - This property should be set to TRUE when navigating to relationship properties or objects.
- **Graph edge name** - The name of the graph edge alias associated with the column item.

The new column set will appear in the *Library* pane.

Manage Column Items

- **Select the new *column item* and click the left arrow button to relate the item to the column set.**

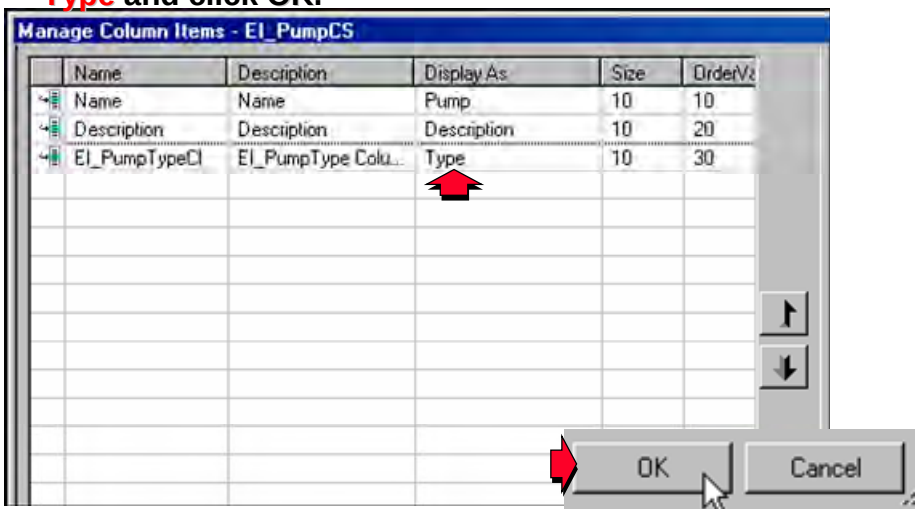


© 2008. Intergraph Corp.
All Rights Reserved.

Once the new column item has been added to the related items pane, the *DisplayAs* field can be modified.

Manage Column Items

- **Change the *Display As* field to to show the label as the Type and click OK.**



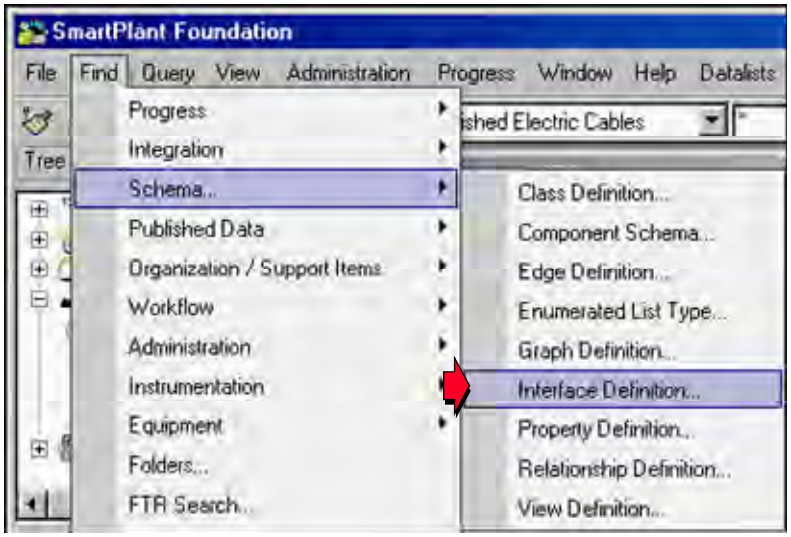
© 2008. Intergraph Corp.
All Rights Reserved.

Now the new column set must be related to the pump object via an interface.

Perform a search to locate the interfaces realized by the pump class.

Manage Related Item

- **From the menu, click *Find > Schema > Interface Definition*.**

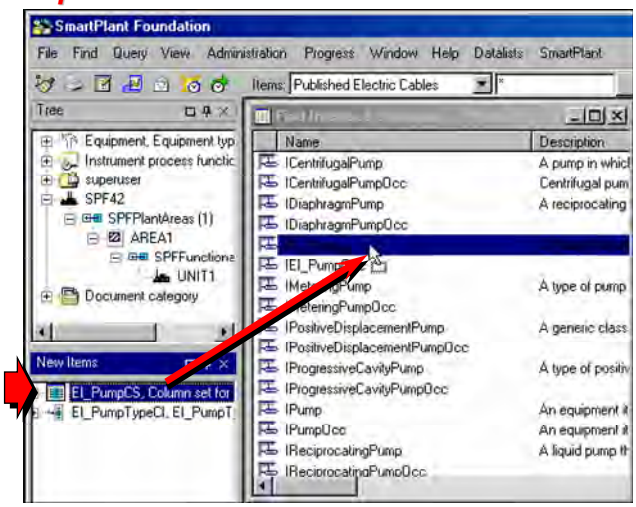


© 2008. Intergraph Corp.
All Rights Reserved.

The interfaces will be displayed in a list view window.

Manage Related Item

- **Drag and drop the column set *EIPumpCS* to relate it to the *IEI_Pump* Interface Definition.**

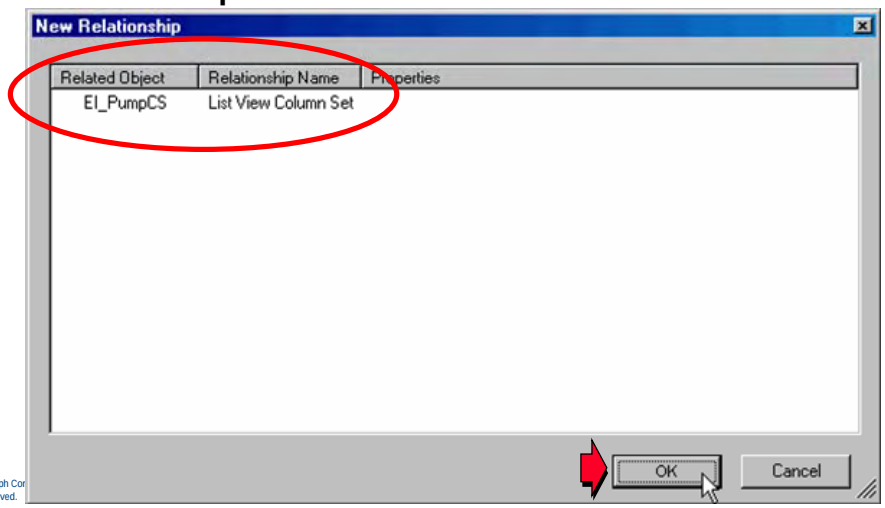


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Relationship* form will appear.

Manage Related Item

- **Click OK to complete the new *List View Column Set* relationship.**

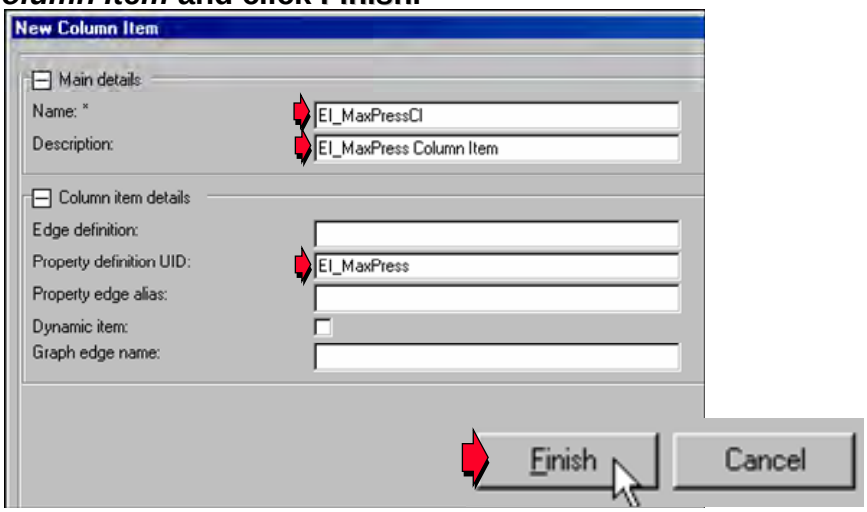


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *Manage Column Items* form to create additional column items. Select the ***Create*** command on the *Library* pane. The *New Column Item* form will appear.

Manage Column Items

- **Enter the appropriate values to define the Max Pressure *Column Item* and click Finish.**



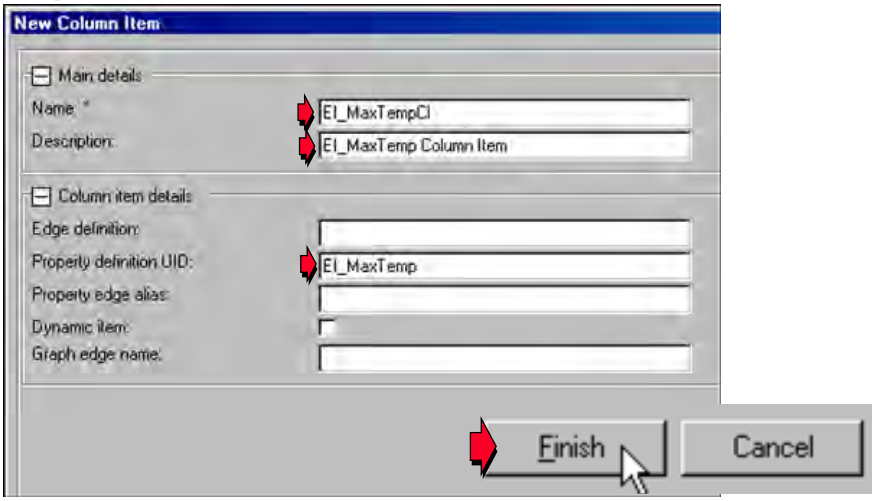
© 2008. Intergraph Corp.
All Rights Reserved.

Repeat these steps to define the remaining desired column items.

Select the ***Create*** command on the *Library* pane. The *New Column Item* form will appear.

Manage Column Items

- **Enter the appropriate values to define the Max Temperature *Column Item* and click Finish.**

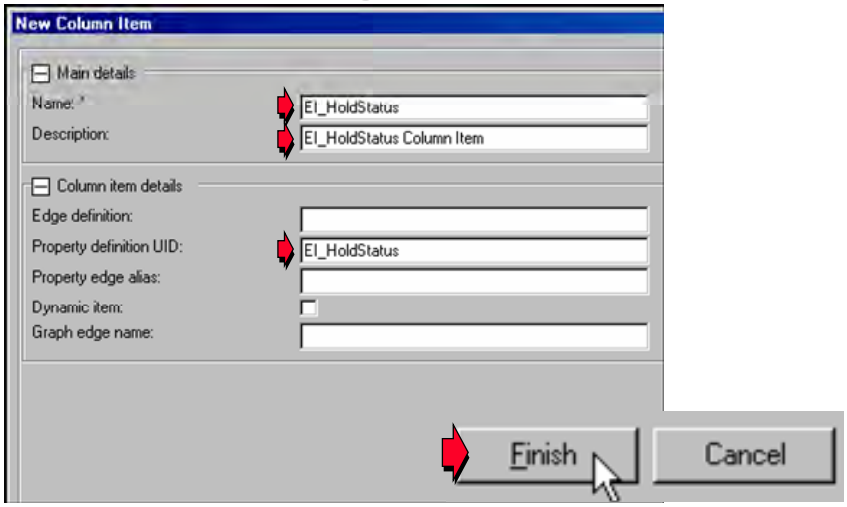


© 2008. Intergraph Corp.
All Rights Reserved.

Repeat again to define more column items. Select the ***Create*** command on the *Library* pane.

Manage Column Items

- **Enter the appropriate values to define the Hold Status *Column Item* and click Finish.**

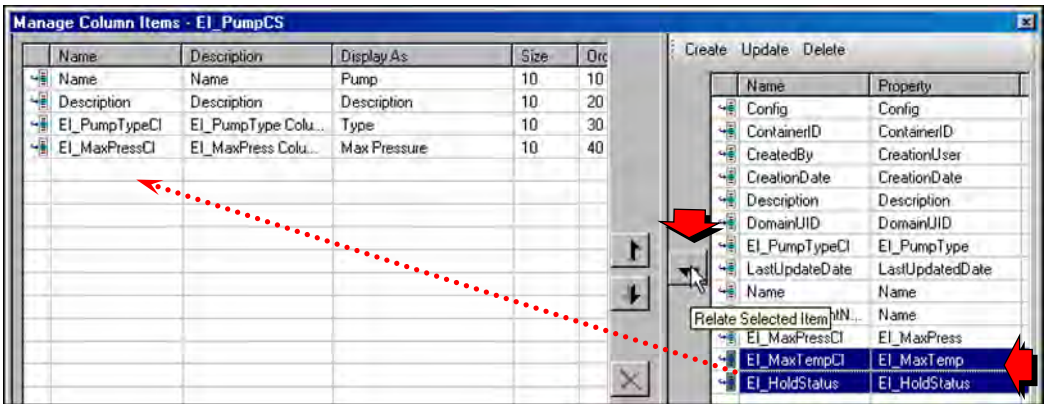


© 2008. Intergraph Corp.
All Rights Reserved.

Relate the new column items to the column set.

Manage Related Item

- **Select the new column items and click the left arrow button to relate the items to the column set.**

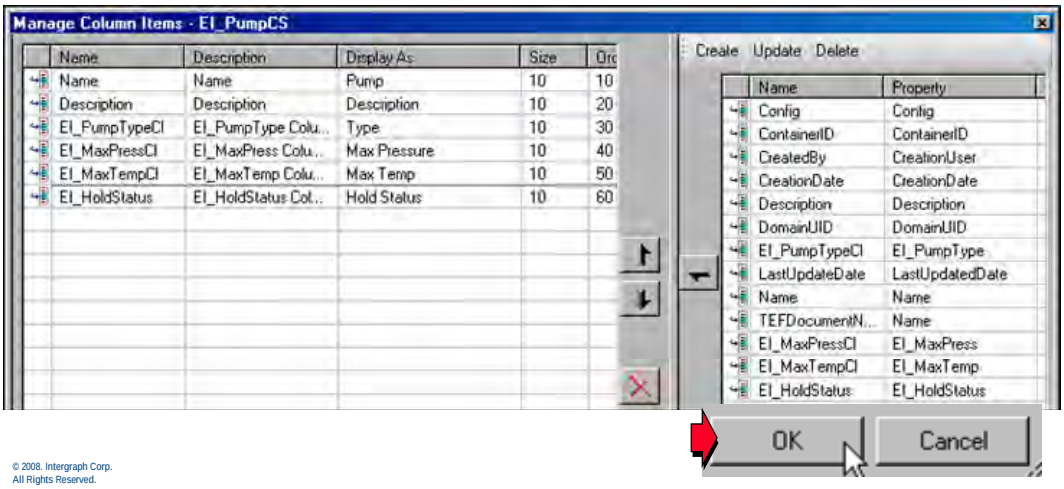


© 2008. Intergraph Corp.
All Rights Reserved.

Click in the box to edit the *DisplayAs* labels.

Manage Column Items

- **Verify that all of the new column items are related to the column set and click OK.**

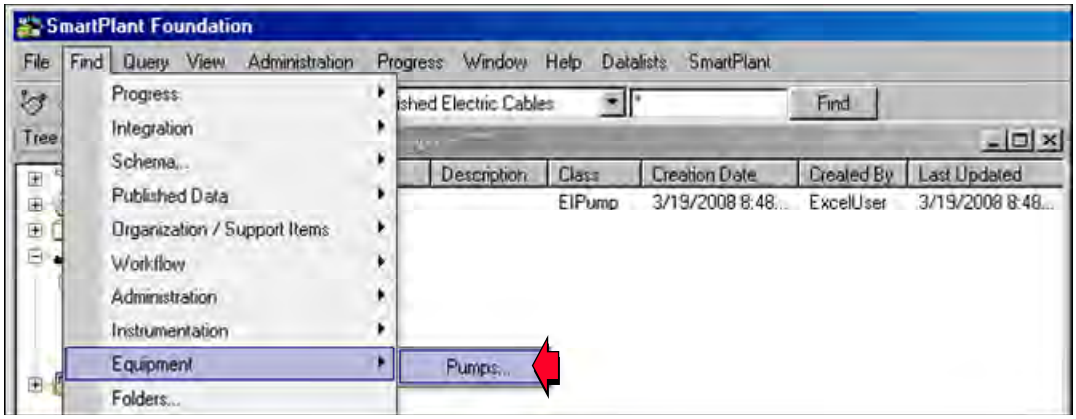


© 2008. Intergraph Corp.
All Rights Reserved.

With the column set and column items configured, perform a search to display pump object instances. This will test the column set configuration.

Manage Column Items

- **From the menu, click *Find > Equipment > Pumps.***



© 2008. Intergraph Corp.
All Rights Reserved.

The custom *Find Pumps* dialog will appear.

Manage Column Items

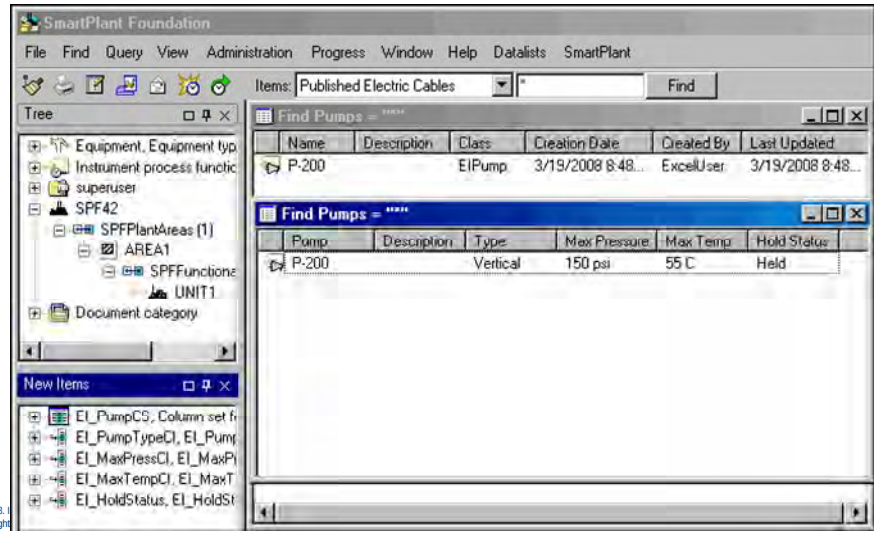
- **Click OK to find all *Pumps*.**

© 2008. Intergraph Corp.
All Rights Reserved.

In the below example, two list view windows are shown, The top list view window uses the default column set configuration but the bottom list view window is shown with the custom column set configuration.

Manage Column Items

Compare the customized column set to the default columns.



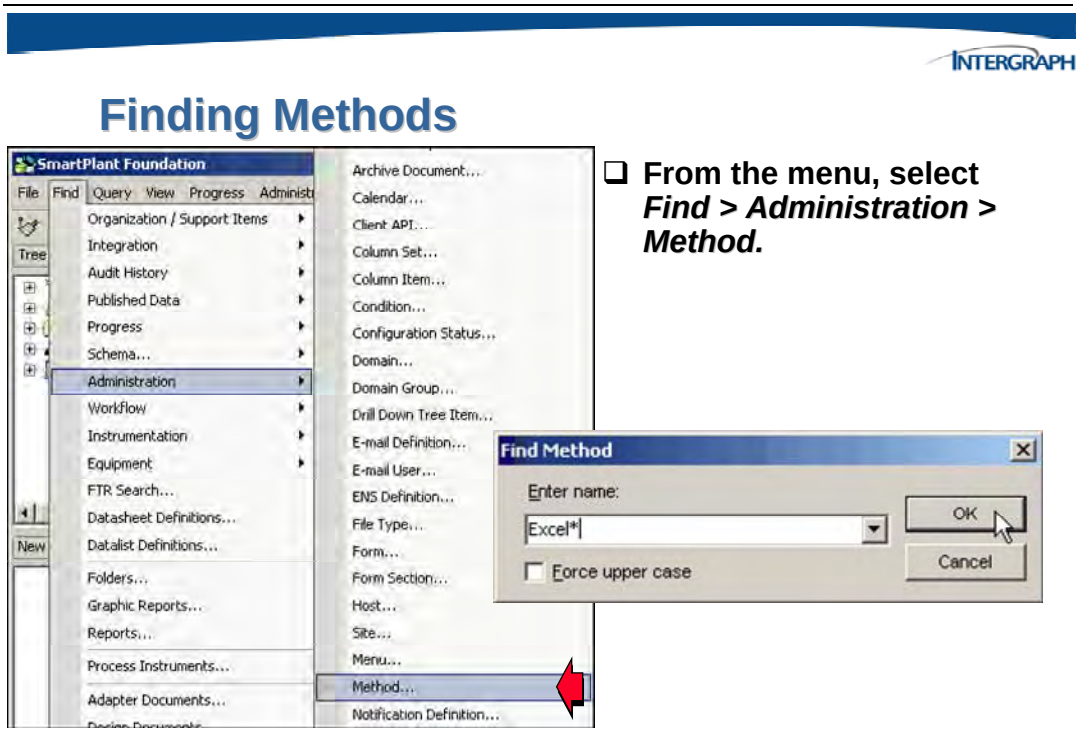
Default

Custom

5.9 Configuring Additional Methods and Menus

Three more methods (*ExcelImportPumpList*, *ExcelImportPumpWithCompare*, *and ExcelImportPumpListNoDocNoCompare*) have been loaded with the DTC Loader and need to be related to an access group.

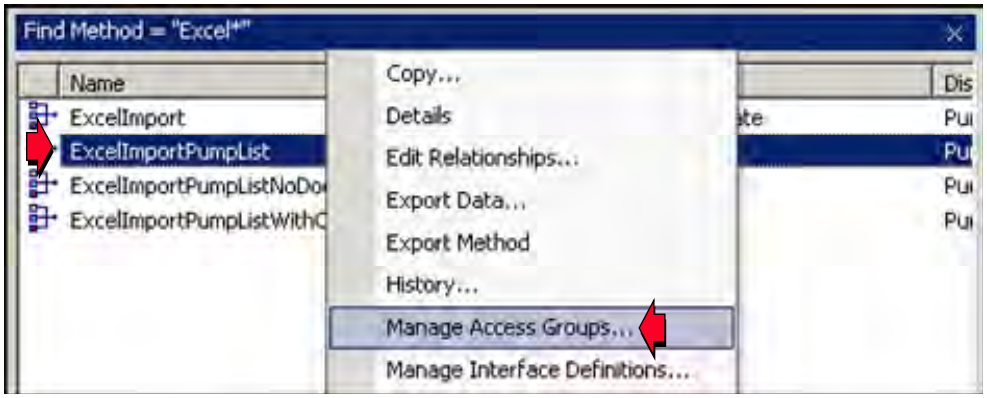
First, perform a find to locate the methods that were loaded.



Enter the value *Excel** in the ***Find Method*** quick find dialog.

Manage Related Items

- **Right click on *ExcelImportPumpList* and from the shortcut menu, select *Manage Access Groups*.**

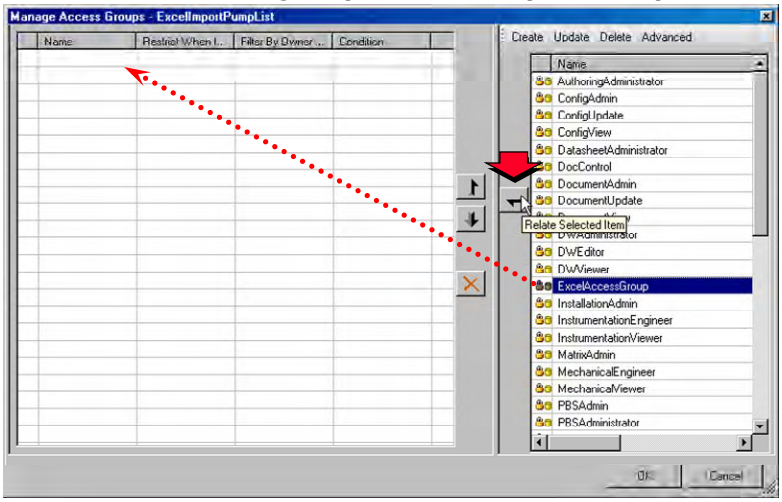


© 2008. Intergraph Corp.
All Rights Reserved.

Relate the **Excel Access Group** (which was created by the load file) to the *ExcelImportPumpList* method.

Manage Related Items

- **Select *ExcelAccessGroup* and click the left arrow button to relate this access group to *ExcelImportPumpList*.**

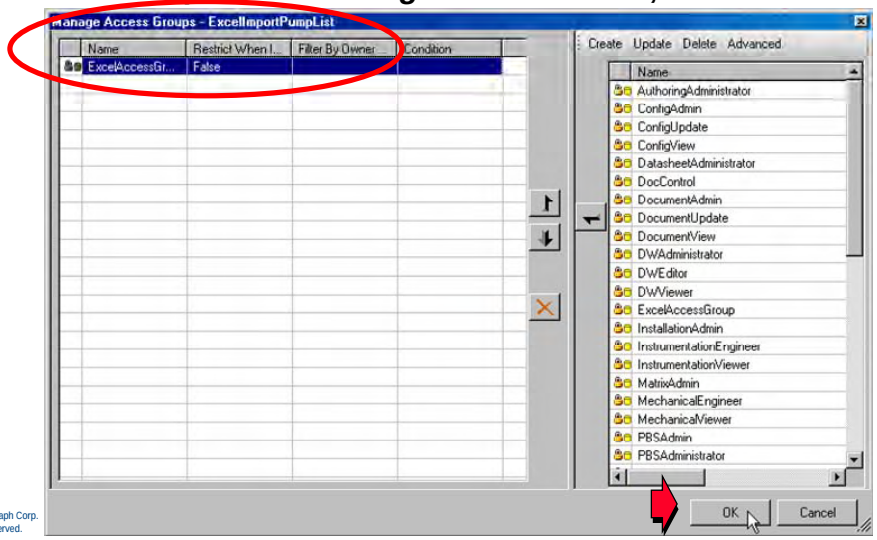


© 2008. Intergraph Corp.
All Rights Reserved.

The update is saved to the database on selection of **OK**.

Manage Related Items

- **To complete the *Manage Related Items*, click OK.**

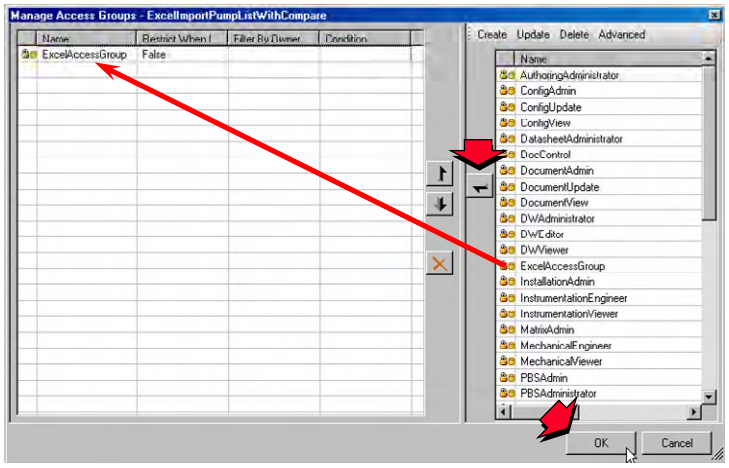


© 2008. Intergraph Corp.
All Rights Reserved.

Right click on *ExcelImportPumpWithCompare* and from the shortcut menu, select ***Manage Access Groups***.

Manage Related Items

- **Select *ExcelAccessGroup* and click the left arrow button to relate this access group to *ExcelImportPumpWithCompare* and click OK.**

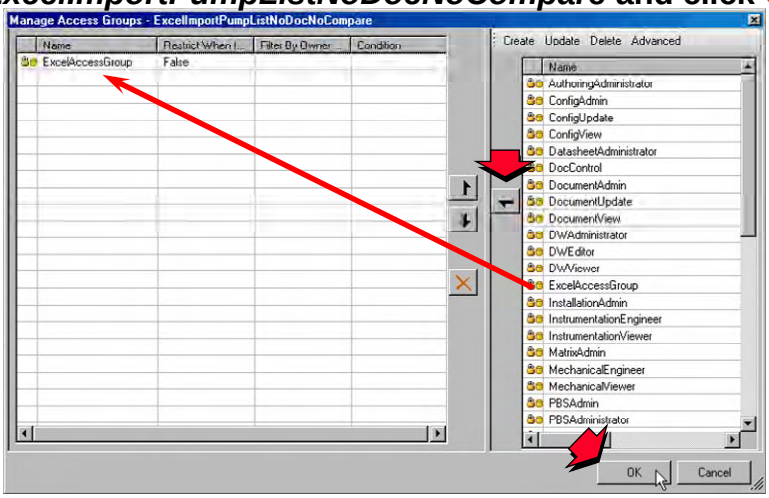


© 2008. Intergraph Corp.
All Rights Reserved.

Right click on *ExcelImportPumpListNoDocNoCompare* and from the shortcut menu, select ***Manage Access Groups***.

Manage Related Items

- **Select *ExcelAccessGroup* and click the left arrow button to relate this access group to *ExcelImportPumpListNoDocNoCompare* and click OK.**

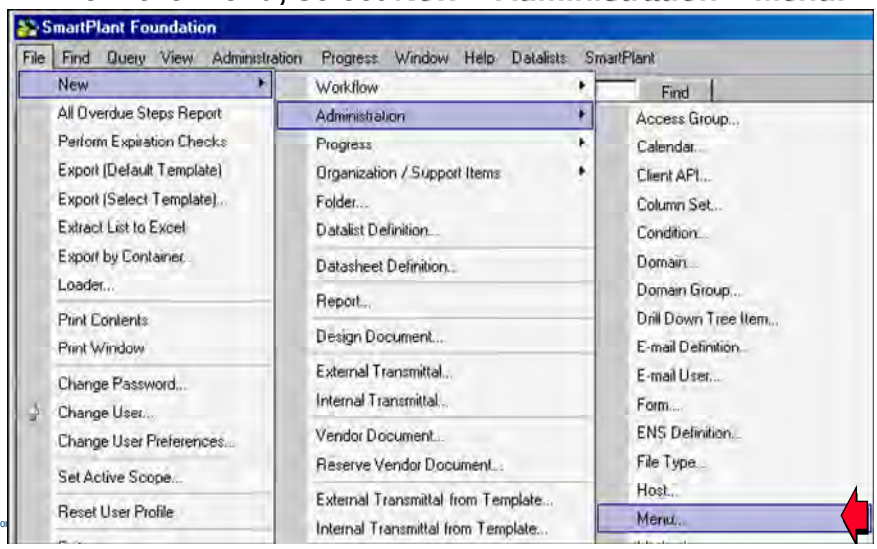


© 2008. Intergraph Corp.
All Rights Reserved.

Next, create a new menu for the methods.

Creating Menus

- **From the menu, select *New > Administration > Menu*.**

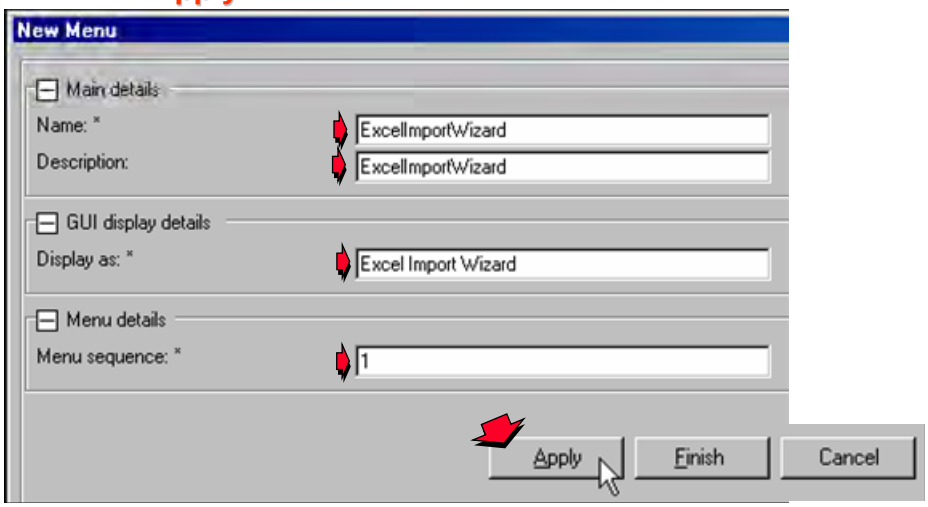


© 2008. Intergraph Co
All Rights Reserved.

The *New Menu* form is displayed.

Creating Menus

- **Enter the values for the new menu, set *Menu sequence* to 1 and click Apply.**



© 2008. Intergraph Corp.
All Rights Reserved.

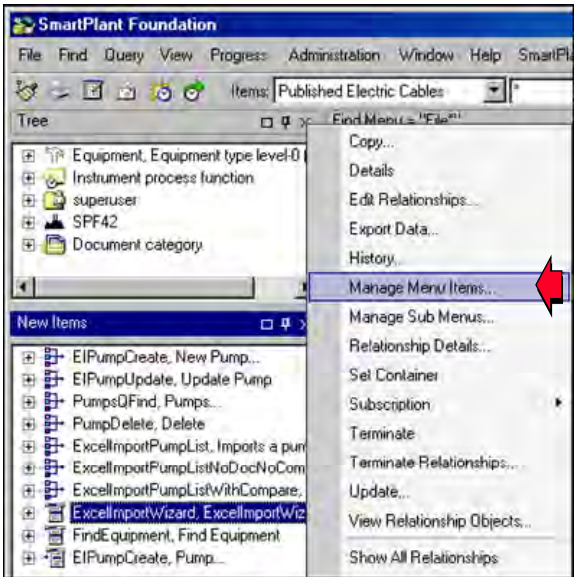
The *New Menu* form contains the following fields:

- **Name** - specifies the name/purpose of the menu and is a required field.
- **Description** - specifies a description for the menu and is an optional field.
- **Display as** - specifies the name that will be displayed on the SmartPlant Foundation Desktop Client menu bar. This is a required field.
- **Menu sequence** - defines a sequence number for the menu node. Sequence numbers define the position of the menu node on the menu going from left to right.

Once menu nodes have been created, the next step is to relate menu items.

Creating Menu Items

- **Right click on *ExcelImportWizard* and from the shortcut menu, select *Manage Menu Items*.**

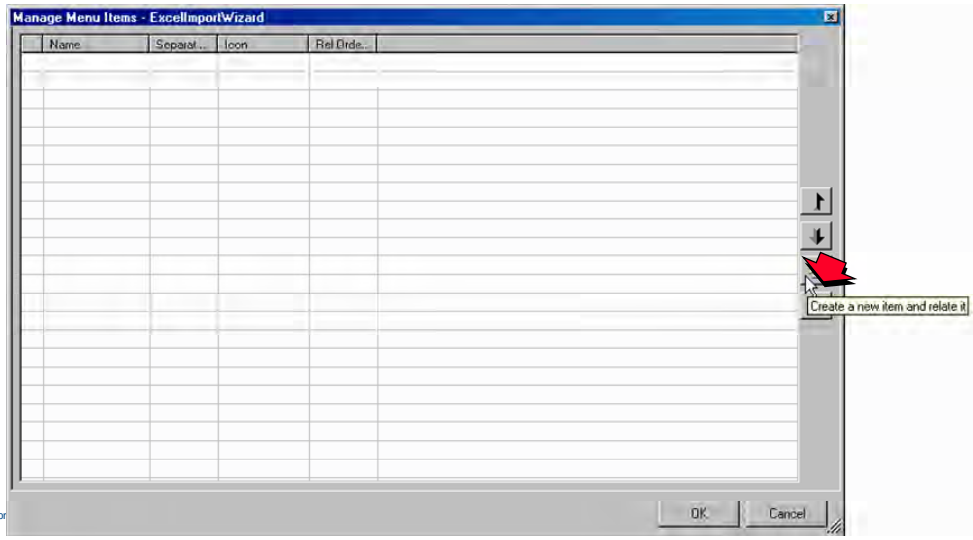


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Menu Items* form will appear.

Creating Menu Items

- **On the *Manage Menu Items* dialog, click the create button.**

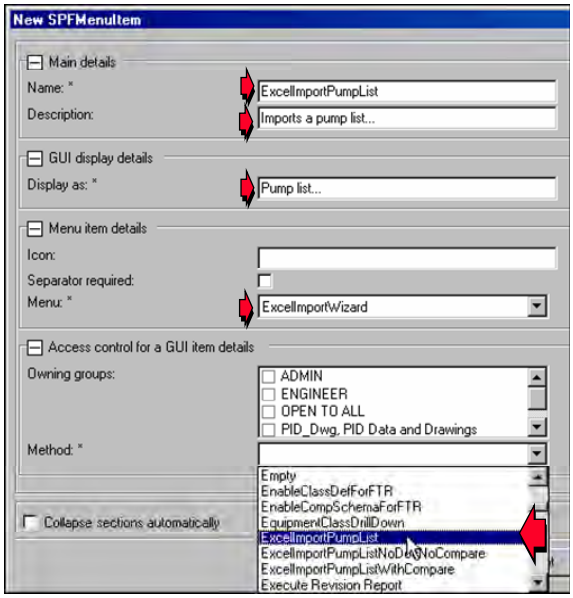


© 2008. Intergraph Cor
All Rights Reserved.

The *New SPFMenuItem* form will appear.

Creating Menu Items

- **Enter the values for the new *MenuItem*.**
- **Set the Method field to *ExcelImportPumpList*.**



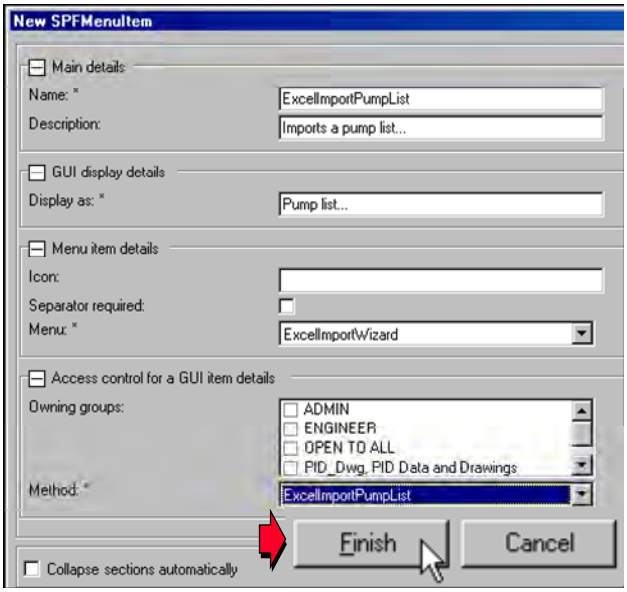
© 2008. Intergraph Corp.
All Rights Reserved.

The *New SPFMenuItem* form contains the following fields:

- **Name** - specifies the name/purpose of the menu item and is a required field.
- **Description** - specifies a description for the menu and is an optional field.
- **Display as** - specifies the name that will be displayed on the SmartPlant Foundation Desktop Client menu. This is a required field.
- **Icon** - associates an icon with the menu item. Icons must be located in the \SmartPlant Foundation 2008 Server Files\Web_Sites\<site name>\Icons folder. Icons must be in .gif format but icons are not typically used for menu items.
- **Separator required** - select the check box to insert a separator between this menu item and other menu items above or below it.
- **Menu** - specifies the menu node that this menu item will be associated with. This is a required field.
- **Owning groups** - select the owning group that can access this menu item. Selecting an owning group will restrict who can access the item. This is not to be confused with access group control that is configured on the related method. This is an optional field.
- **Method** - select a method from the list. The method must have an access group and is a required field.

Creating Menu Items

- **To complete the new menu item, click Finish.**

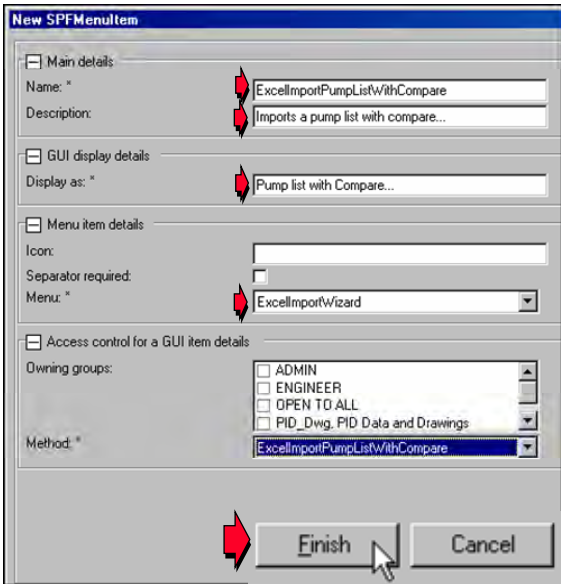


© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the steps to create another menu item. Start from the *Manage Menu Items* dialog by clicking the **Create** button.

Creating Menu Items

- **Create a another new menu item called *ExcelImportPumpList WithCompare* then click Finish.**



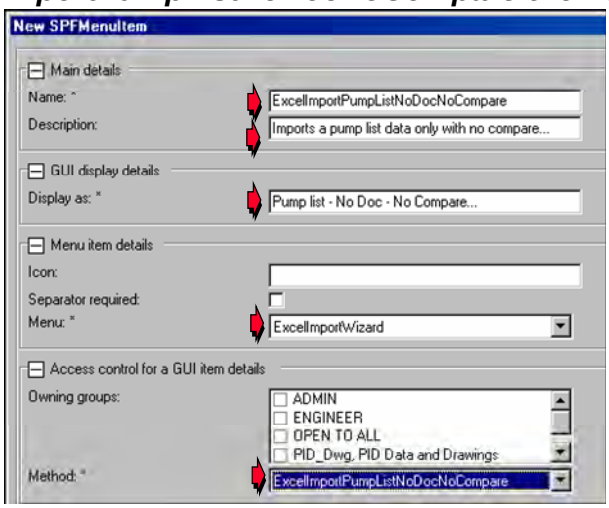
© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the steps to create the last menu item needed.

Start from the *Manage Menu Items* dialog by clicking the **Create** button.

Creating Menu Items

- **Create a another new menu item called *ExcelImportPumpListNoDocNoCompare* then click Finish.**

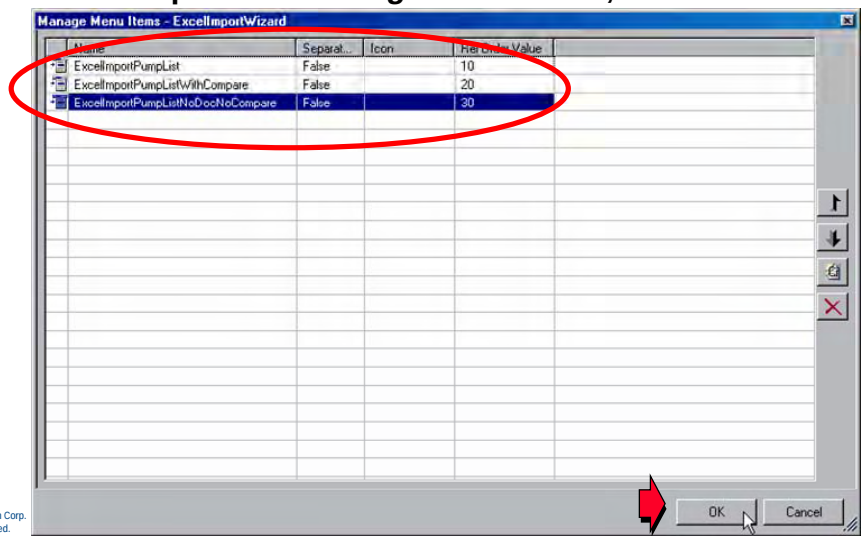


© 2008. Intergraph Corp.
All Rights Reserved.

These three menu items will be related to the *ExcelImportWizard* menu node.

Creating Menu Items

- **To complete the *Manage Menu Items*, click OK.**



© 2008. Intergraph Corp.
All Rights Reserved.

Perform a search for the *File* menu node in order to create a set of sub menus, very similar to the *New* sub menu.

Managing Sub Menus

- **Enter *File* with the wildcard character, *, to find and display all Menus that start with *File* and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

The new sub menu structure will look like this:

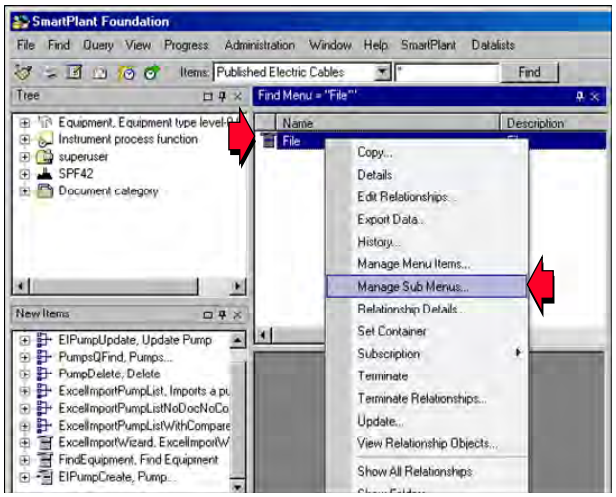
File (Menu node)

- ExcelImportWizard (Sub menu)
  - ExcelImportPumpList (Menu item)
  - ExcelImportPumpListWithCompare (Menu item)
  - ExcelImportPumpListNoDocNoCompare (Menu item)

The existing *File* menu node will display in the list view browser.

Managing Sub Menus

- **Right click on the *File* menu and from the shortcut menu, select *Manage Sub Menus.***

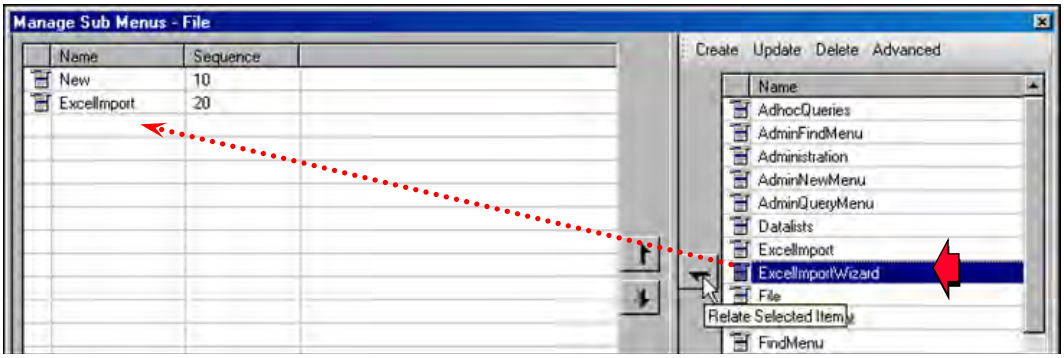


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Sub Menus* form can be used to manage the sub menu (that is create, delete, and change the sequence of the sub menu).

Managing Sub Menus

- **Select *ExcelImportWizard* and click the left arrow button to relate the submenu to the *File* menu.**

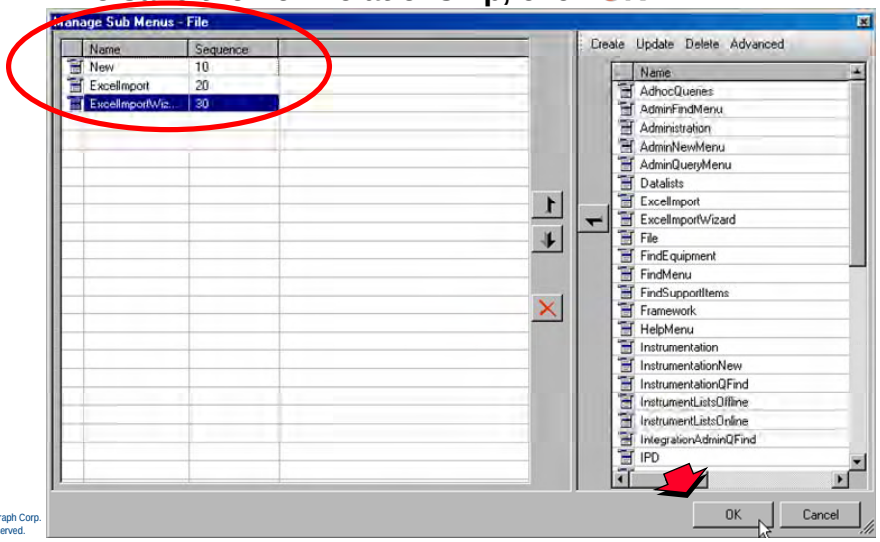


© 2008. Intergraph Corp.
All Rights Reserved.

The *ExcelImportWizard* menu node was created in an earlier step.

Managing Sub Menus

- **To save the new relationship, click OK.**



© 2008. Intergraph Corp.
All Rights Reserved.

5.10 Enumeration Lists and Hierarchies

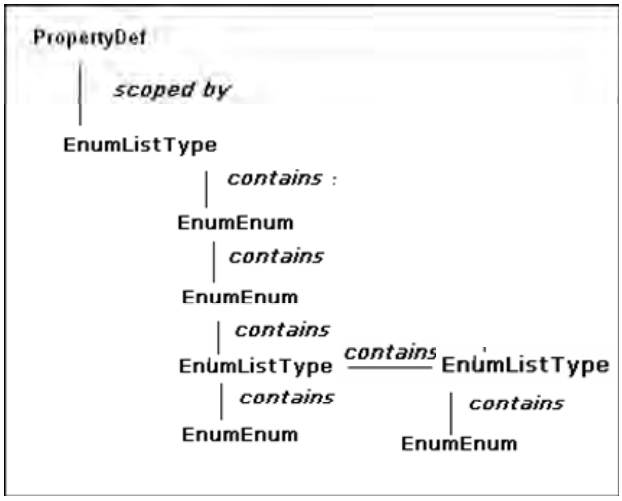
To define lookups for a property, *enumerated lists* are created and the property scoped by the enumerated list.

Enumerated lists can contain all the possible enumerated entries for a property definition, or they can be combined to form an enumeration hierarchy. When enumerated lists are combined in this way, they create child enumerated lists that are referenced by other enumerated lists.

When this happens, a special property type (enumerated list level type) can be used to describe all of the enumerations at a given level in the enumeration hierarchy. When a property definition has an associated enumerated list level type, it effectively allows dependent pick lists to be defined.

Enumeration Hierarchies

All *EnumListTypes* contain EnumEnums (list entries) or other EnumListTypes (enumerated lists).



© 2008. Intergraph Corp.
All Rights Reserved.

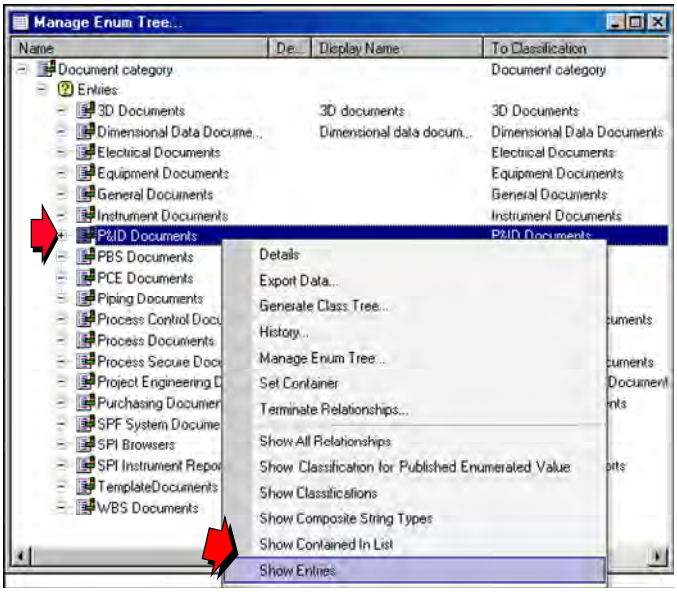
Enumerated Lists, *Enumeration Hierarchies* and *Enumerated List Level Types* must be created using the **Schema Editor** and then loaded into the SPF database.

Once these objects have been loaded, they can be viewed using the Desktop Client.

In this example, a search has been performed to display an Enumeration Hierarchy, and enumerated list named *Document category*.

Enumeration Hierarchies

- **Right click on the P&ID Documents entry and select *Show Entries* from the shortcut menu.**

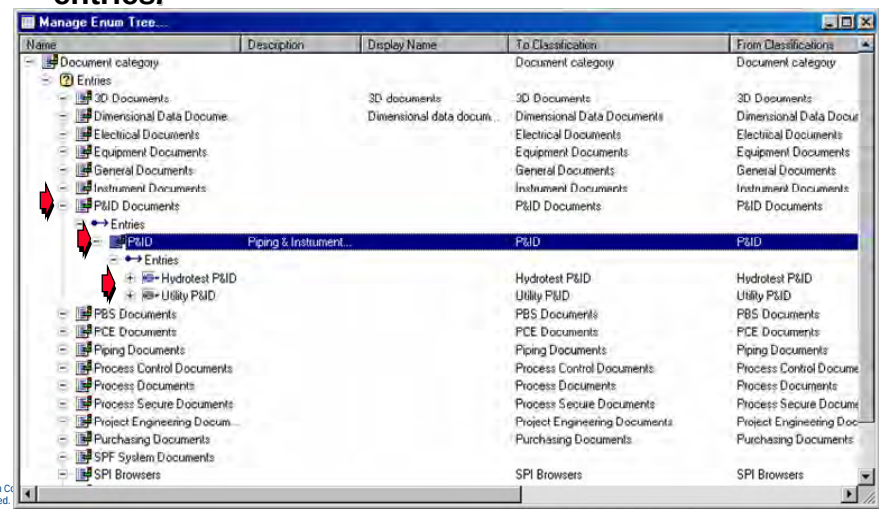


© 2008. Intergraph Corp.
All Rights Reserved.

The enumerated list type will expand to show another related enumerated list type.

Enumeration Hierarchies

- **Expand the enumeration hierarchy to view all of the related entries.**

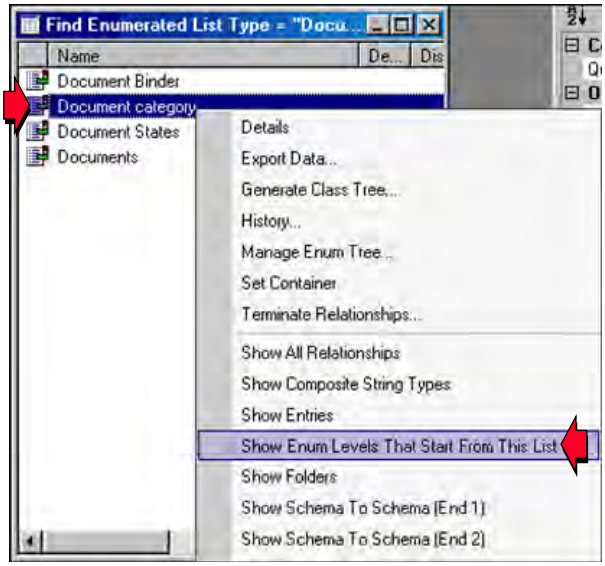


© 2008. Intergraph Co
All Rights Reserved.

Since this enumerated list type is used as a classification tree in the Desktop Client, it also uses enumerated list level types.

Enumeration Hierarchies

- **Right click on the Document category Enumerated List Type and from the shortcut menu, select *Show Enum Levels That Start From This List.***



© 2008. Intergraph Corp.
All Rights Reserved.

Enumeration Hierarchies

- **Review the Enum Levels related to the top level *Enumerated List Type*, Document category.**

© 2008. Intergrap
All Rights Reserv

5.11 Activity 3 – Creating Column Sets

Complete **Chapter 5 - Activity 3** in the SPF Configuration and Administration II activity workbook.

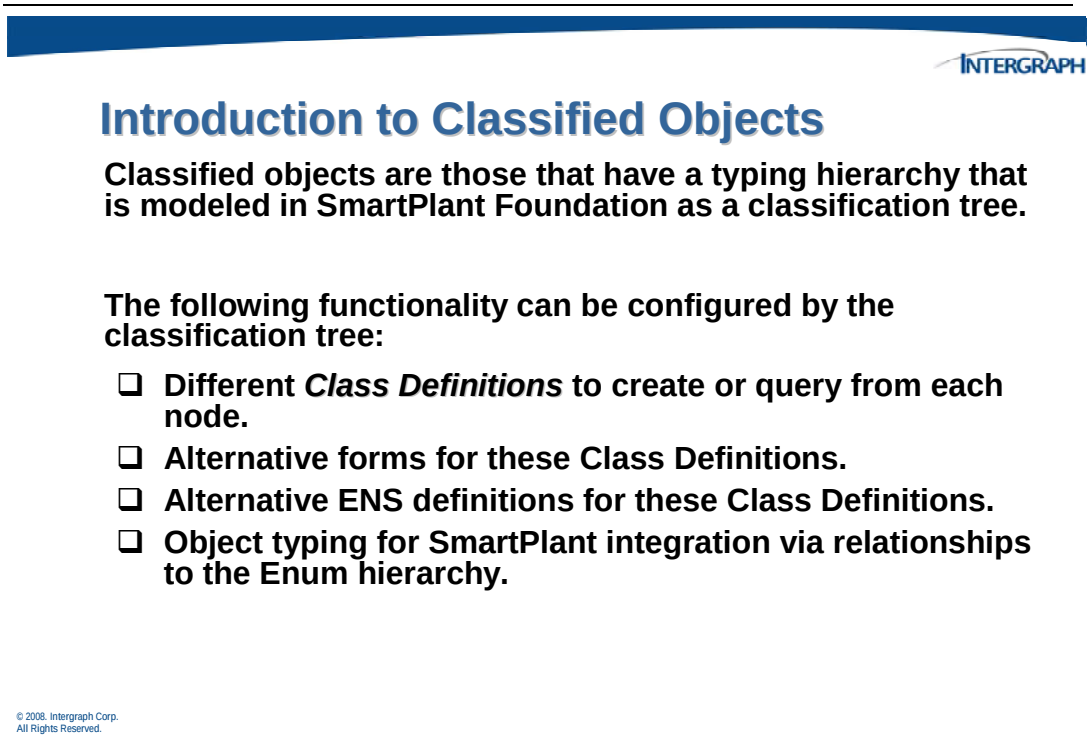
CHAPTER 6

Creating Classified Objects

6. Introduction to Classified Objects

This chapter will cover the details of how to configure Classified Objects and a Classified Object Tree.

Classified objects are those that have a typing hierarchy that are modeled in SmartPlant Foundation as a classification tree. The classification tree consists of first class objects so that they support relationship management and shortcut menus to the objects that they classify. They can be configuration controlled enabling different hierarchies in different configurations.



The classification tree is similar to that used in the Business Object model in SmartPlant Foundation 3.x, but in SmartPlant Foundation 4.x it is also used to expose the SmartPlant integration typing hierarchies. In the SmartPlant Schema, there are Enum typing hierarchies for documents and tags but this is a rigid structure that cannot be easily modified and rearranged or sub selected for use in projects.

In SmartPlant Foundation 4.x, the Enum hierarchies are hidden behind, and related to, a classification tree. The classification trees need a related Enum hierarchy only to support SmartPlant integration as it is the Enum typing hierarchies that are used by the tools but they are still exposed as classification trees in SmartPlant Foundation for consistency and GUI support.

Interactive business item creation and query is performed in SmartPlant Foundation using the classification tree with the Enum typing set on the objects in the background. SmartPlant integration still publishes documents and tags based on the Enum typing and when the objects are loaded into SmartPlant Foundation, relationships to the classification

tree are created on the fly. There are two separate relationships, one mapping from the classification tree to the Enum hierarchy for interactive use and one mapping from the Enum hierarchy to the classification tree for use by the publish process.

Introduction to Classified Objects

Each classification tree can potentially classify many business items. For example, one document classification tree can be used to classify many different document classes.

Each tree is usually composed of related objects of the same Class Definition, though that is not a restriction.

To control the access to these different "families" of objects, each of these families are configured using a specific relationship to the Class Definitions that need to be created or queried within that family.

© 2008. Intergraph Corp.
All Rights Reserved.

There may be more than one *family* of objects that follow the same classification such as tags, assets, models, and variants. To control the access to these different *families* of objects, each of these families is configured using a specific relationship to the Class Definitions that need to be created or queried within that family. Not every node of the tree needs these relationships as this configuration is inherited down the tree. To identify a node of the tree as being valid for a particular "family", each "family" is usually represented by an optional interface on the classification tree and each node of the classification tree can then instantiate the relevant interfaces. For example, some nodes may be valid for assets and others for assets, models, and variants. The relevant ClassDefs for each member of the "family" are related to the node of the tree and this relationship is inherited down the tree.

Introduction to Classified Objects

When a "family" uses the same ClassDef on all nodes of the tree, then this relationship only needs to be created at the top node and the tree can be described as a Homogeneous tree as is the case for the Design Document.

When the "family" uses different ClassDefs at different nodes of the tree, it can be described as Heterogeneous as is the case for equipment with different ClassDefs for pumps, vessels, and so forth.

© 2008. Intergraph Corp.
All Rights Reserved.

6.1 Modeling the Classification

Use the *Schema Editor* to model classification trees to describe the breakdown of documents, tags, assets, etc., in the system in terms of the customer's structure or their client's structure.

The classification tree is created as a hierarchy of related objects with optional interfaces instantiated to manage the creation and query of classified objects.

The classification trees are modeled with the following interfaces as described in the Classified Object Model section in the *SmartPlant Foundation How to Configure Classified Objects* document.

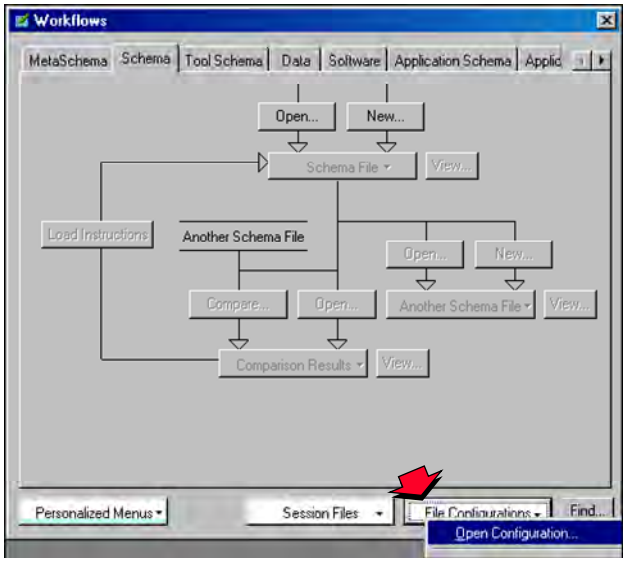
- Primary Interface (required) for example, *IEngineeringDocClass*.
- ISPFObjClass (required) defines this as a classification object.
- ISPFSubscribableItem (required) to support user subscription to the class trees. If this is not needed, then this interface does not need to be realized.
- ISPFDisplayOverride (optional). If this is not needed, then this interface does not need to be realized.

Start the Schema Editor and open a configuration file.

INTERGRAPH

Modeling Classified Objects

- **Start the Schema Editor and click File Configurations to open a configuration file.**



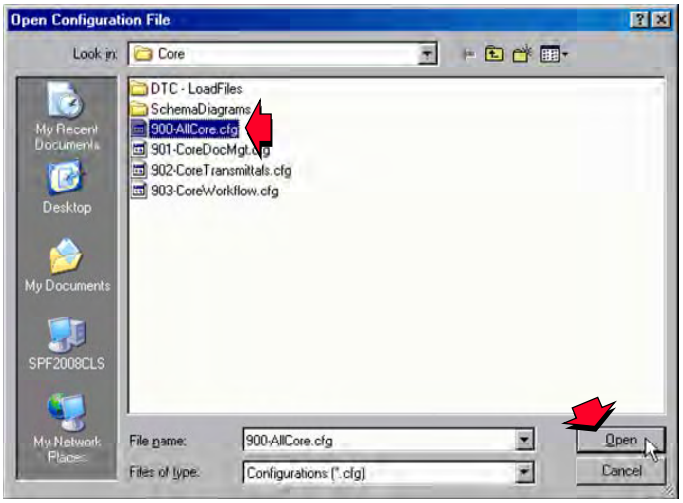
© 2008. Intergraph Corp.
All Rights Reserved.

Open the core schemas as well as the extension schema containing the custom classified class definitions.

The location for the core schema files is **\SmartPlant\Foundation\2008\Models\Core**.

Modeling Classified Objects

- **Choose *900-AllCore.cfg* from the Core folder and click Open.**

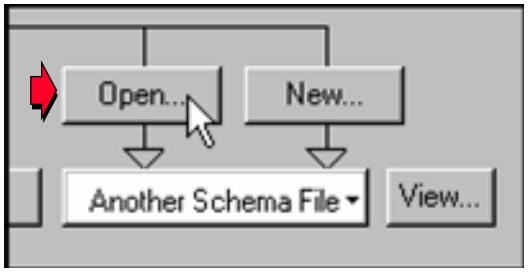


© 2008. Intergraph Corp.
All Rights Reserved.

Once the core schema files have been opened with the *900-AllCore.cfg*, the extension schema file (01-SCHEMA-EngDocs.xml) will need to opened.

Modeling Classified Objects

- **Open the schema file containing the *EngineeringDocument* definitions by clicking the Open button above *Another Schema File.***

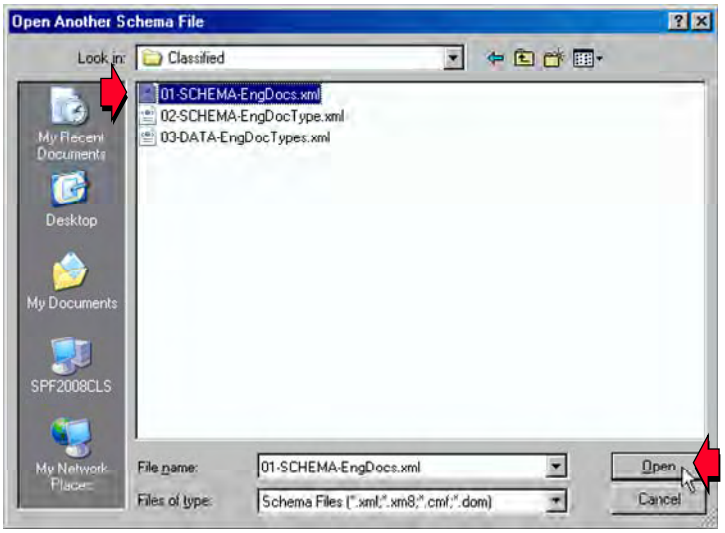


© 2008. Intergraph Corp.
All Rights Reserved.

The extension schema for this example is located in the *D:\SPF42_Training\Classified* folder.

Modeling Classified Objects

- **Select *01-SCHEMA-EngDocs.xml* and click Open.**

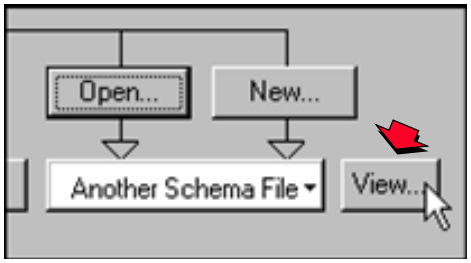


© 2008. Intergraph Corp.
All Rights Reserved.

After the needed schema files have been opened, use the **View** button to begin to model the classified object classes.

Modeling Classified Objects

- **In the *Workflows* dialog box click the View button (next to *Another Schema File) to* View the schema.**

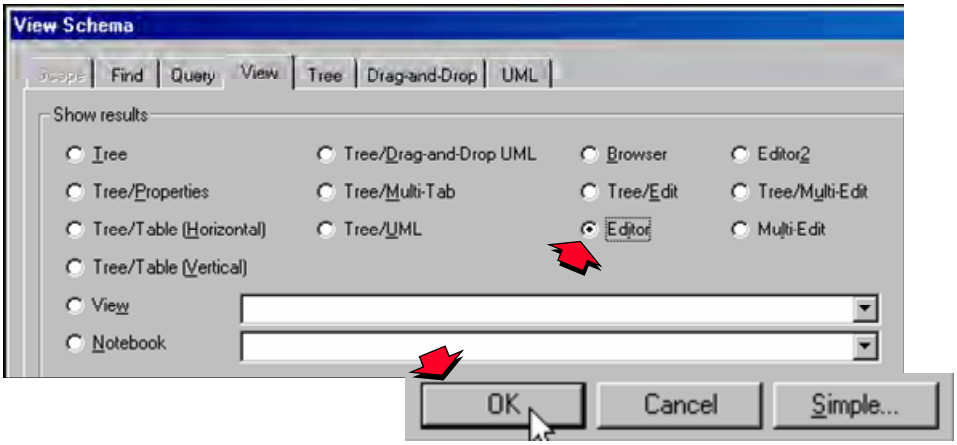


© 2008. Intergraph Corp.
All Rights Reserved.

When the *View Schema* dialog displays, choose the type of view to use for this editing session.

Modeling Classified Objects

- **Choose the *Editor* option and click OK.**

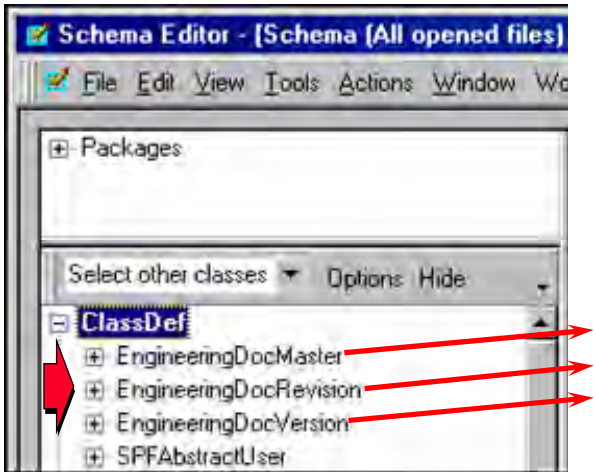


© 2008. Intergraph Corp.
All Rights Reserved.

This example shows the three-tier document structure that has already been modeled for use in configuring a classified objects tree.

Modeling Classified Objects

- **Drag and drop the classified *ClassDefs* from the tree into the right pane.**
- **When prompted, create a custom view.**

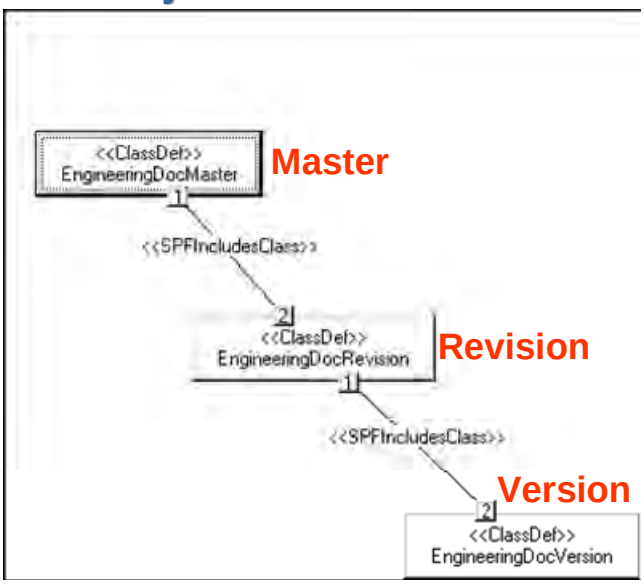


© 2008. Intergraph Corp.
All Rights Reserved.

These three documents will be related to the classified objects tree which will be shown later in this chapter.

Modeling Classified Objects

This UML diagram shows the three tier document structure.

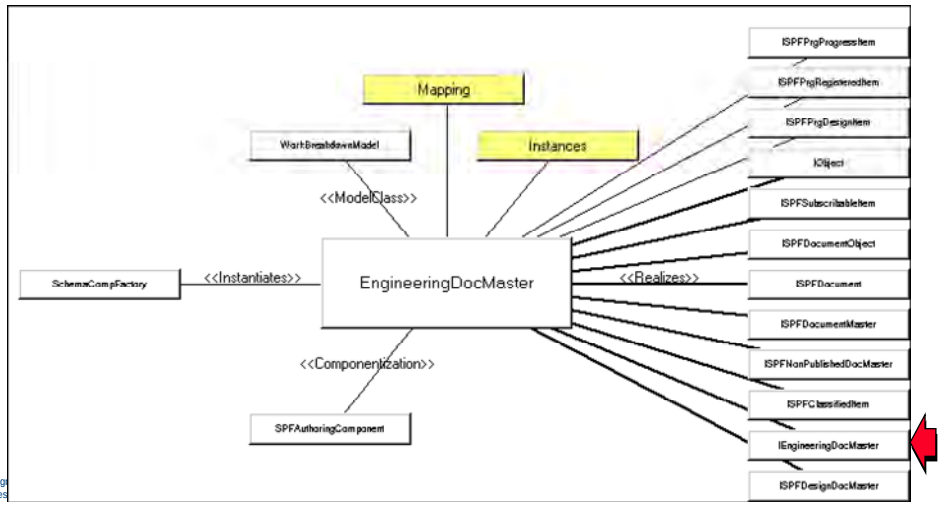


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *View* or *Edit* command to see the structure of the document master Class Def.

Modeling Classified Objects

This UML diagram shows the EngineeringDocMaster document structure.

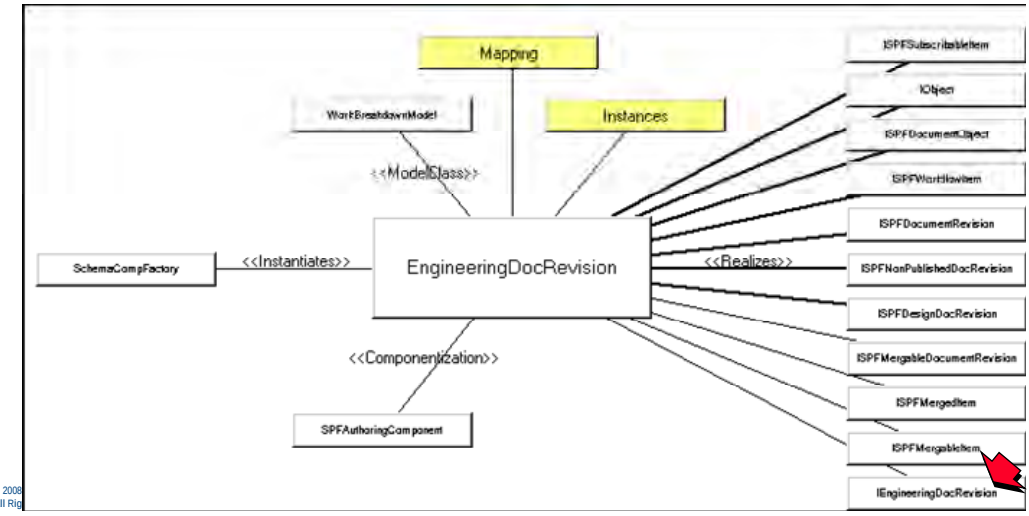


© 2008. Intergr
All Rights Res

Use the *View* or *Edit* command to see the structure of the document revision Class Def.

Modeling Classified Objects

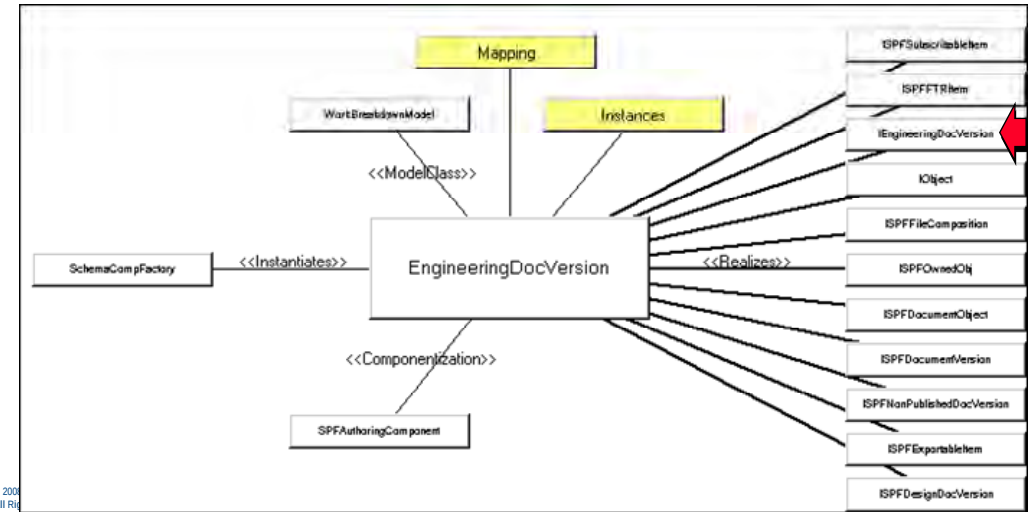
This UML diagram shows the EngineeringDocRevision document structure.



Use the *View* or *Edit* command to see the structure of the document version Class Def.

Modeling Classified Objects

This UML diagram shows the EngineeringDocVersion document structure.



Below is the **01-SCHEMA-EngDocs.xml** that creates the *classes*, *interfaces* and *properties* for the **EngineeringDocument** classified object.


```
- <Container CompSchema="MetaSchema" Scope="Schema" SoftwareVersion="04.02.00.04"
IObject_ContainerID="Core.SPFDesignDocumentSchema" IObject_OBID="Auto"
IObject_CreationDate="Now" IObject_TerminationDate="9999/12/31-23:59:59:999"
IObject_LastUpdatedDate="Now" IObject_CreationUser="superuser"
IObject_DomainUID="Auto" IObject_UniqueKey="Auto" IRel_DomainUID1="Auto"
IRel_DomainUID2="Auto">

- <ClassDef>

  <IObject UID="EngineeringDocMaster" Name="EngineeringDocMaster"
Description="Engineering Document" />

  <IClassDef />

  <ISchemaObj DisplayName="Engineering document" />

  <ISPFClassDefExt SPFIsConfigurationControlled="True" SPFUniqueKeyDef="EDM,Name" />

  </ClassDef>

- <Rel>

  <IObject UID="ModelClass.EngineeringDocMaster" />

  <IRel UID1="WorkBreakdownModel" UID2="EngineeringDocMaster" DefUID="ModelClass"
OrderValue="1" IsRequired="False" />

  <ISchemaObj />

  </Rel>

- <Rel>

  <IObject UID="Instantiates.EngineeringDocMaster" />

  <IRel UID1="SchemaCompFactory" UID2="EngineeringDocMaster" DefUID="Instantiates"
OrderValue="1" IsRequired="False" />

  <ISchemaObj />

  </Rel>

- <Rel>

  <IObject UID="Componentization.EngineeringDocMaster" />

  <IRel UID1="SPFAuthoringComponent" UID2="EngineeringDocMaster"
DefUID="Componentization" OrderValue="0" IsRequired="False" />

  <ISchemaObj />

  </Rel>

- <InterfaceDef>

  <IObject UID="IEngineeringDocMaster" Name="IEngineeringDocMaster"
Description="Engineering Document Master" />

  <IInterfaceDef />

  <IInterfaceRole />

  <IPropCategory />

  <IEnumMetadata />
```

```
  <ISchemaObj DisplayName="Engineering document master" />
  </InterfaceDef>
- <Rel>
  <IObject UID="EngineeringDocMaster.IEngineeringDocMaster.PI" />
  <IRel UID1="EngineeringDocMaster" UID2="IEngineeringDocMaster"
DefUID="PrimaryInterface" OrderValue="1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="IEngineeringDocMaster.ISPFDesignDocMaster" />
  <IRel UID1="IEngineeringDocMaster" UID2="ISPFDesignDocMaster" DefUID="Implies"
OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.IObject" />
  <IRel UID1="EngineeringDocMaster" UID2="IObject" DefUID="Realizes" OrderValue="0"
IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.ISPFSubscribableItem" />
  <IRel UID1="EngineeringDocMaster" UID2="ISPFSubscribableItem" DefUID="Realizes"
OrderValue="5" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.ISPFDocumentObject" />
  <IRel UID1="EngineeringDocMaster" UID2="ISPFDocumentObject" DefUID="Realizes"
OrderValue="30" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.ISPFDocument" />
  <IRel UID1="EngineeringDocMaster" UID2="ISPFDocument" DefUID="Realizes"
OrderValue="50" IsRequired="True" />
  <ISchemaObj />
  </Rel>
```

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFDocumentMaster" />

<IRel UID1="EngineeringDocMaster" UID2="ISPFDocumentMaster" DefUID="Realizes" OrderValue="60" IsRequired="True" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFNonPublishedDocMaster" />

<IRel UID1="EngineeringDocMaster" UID2="ISPFNonPublishedDocMaster" DefUID="Realizes" OrderValue="80" IsRequired="True" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFClassifiedItem" />

<IRel UID1="EngineeringDocMaster" UID2="ISPFClassifiedItem" DefUID="Realizes" OrderValue="90" IsRequired="True" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFDesignDocMaster" />

<IRel UID1="EngineeringDocMaster" UID2="ISPFDesignDocMaster" DefUID="Realizes" OrderValue="100" IsRequired="True" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.IEngineeringDocMaster" />

<IRel UID1="EngineeringDocMaster" UID2="IEngineeringDocMaster" DefUID="Realizes" OrderValue="100" IsRequired="True" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFPrgProgressItem" />

<IRel UID1="EngineeringDocMaster" UID2="ISPFPrgProgressItem" DefUID="Realizes" IsRequired="False" />

<ISchemaObj />

</Rel>

- <Rel>

<IObject UID="EngineeringDocMaster.ISPFPrgDesignItem" />

```
  <IRel UID1="EngineeringDocMaster" UID2="ISPFPrgDesignItem" DefUID="Realizes"
IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.ISPFPrgRegisteredItem" />
  <IRel UID1="EngineeringDocMaster" UID2="ISPFPrgRegisteredItem" DefUID="Realizes"
IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.ISPFAProgressDocMaster" />
  <IRel UID1="EngineeringDocMaster" UID2="ISPFAProgressDocMaster" DefUID="Realizes"
IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <ClassDef>
  <IObject UID="EngineeringDocRevision" Name="EngineeringDocRevision"
Description="Design Document Revision" />
  <IClassDef />
  <ISchemaObj DisplayName="Design document revision" />
  <ISPFClassDefExt SPFIsConfigurationControlled="True"
SPFUniqueKeyDef="PARENT,SPFMajorRevision,SPFMinorRevision" />
  </ClassDef>
- <Rel>
  <IObject UID="ModelClass.EngineeringDocRevision" />
  <IRel UID1="WorkBreakdownModel" UID2="EngineeringDocRevision"
DefUID="ModelClass" OrderValue="1" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="Instantiates.EngineeringDocRevision" />
  <IRel UID1="SchemaCompFactory" UID2="EngineeringDocRevision" DefUID="Instantiates"
OrderValue="1" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="Componentization.EngineeringDocRevision" />
```

```
  <IRel UID1="SPFAuthoringComponent" UID2="EngineeringDocRevision"
DefUID="Componentization" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <InterfaceDef>
  <IObject UID="IEngineeringDocRevision" Name="IEngineeringDocRevision"
Description="Engineering Document Revision" />
  <IInterfaceDef />
  <IInterfaceRole />
  <IPropCategory />
  <IEnumMetadata />
  <ISchemaObj DisplayName="Engineering document revision" />
  </InterfaceDef>
- <Rel>
  <IObject UID="EngineeringDocRevision.IEngineeringDocRevision.PI" />
  <IRel UID1="EngineeringDocRevision" UID2="IEngineeringDocRevision"
DefUID="PrimaryInterface" OrderValue="1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="IEngineeringDocRevision.ISPFDesignDocRevision" />
  <IRel UID1="IEngineeringDocRevision" UID2="ISPFDesignDocRevision" DefUID="Implies"
OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.IObject" />
  <IRel UID1="EngineeringDocRevision" UID2="IObject" DefUID="Realizes" OrderValue="0"
IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFDocumentObject" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFDocumentObject" DefUID="Realizes"
OrderValue="20" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
```

```
  <IObject UID="EngineeringDocRevision.ISPFWorkflowItem" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFWorkflowItem" DefUID="Realizes"
OrderValue="30" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFDocumentRevision" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFDocumentRevision" DefUID="Realizes"
OrderValue="40" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFNonPublishedDocRevision" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFNonPublishedDocRevision"
DefUID="Realizes" OrderValue="50" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFDesignDocRevision" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFDesignDocRevision" DefUID="Realizes"
OrderValue="60" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFSubscribableItem" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFSubscribableItem" DefUID="Realizes"
OrderValue="-1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFMergableDocumentRevision" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFMergableDocumentRevision"
DefUID="Realizes" IsRequired="False" OrderValue="100" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFMergedItem" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFMergedItem" DefUID="Realizes"
IsRequired="False" OrderValue="110" />
```

```
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.ISPFMergableItem" />
  <IRel UID1="EngineeringDocRevision" UID2="ISPFMergableItem" DefUID="Realizes"
IsRequired="False" OrderValue="120" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.IEngineeringDocRevision" />
  <IRel UID1="EngineeringDocRevision" UID2="IEngineeringDocRevision" DefUID="Realizes"
IsRequired="False" OrderValue="130" />
  <ISchemaObj />
  </Rel>
- <ClassDef>
  <IObject UID="EngineeringDocVersion" Name="EngineeringDocVersion"
Description="Engineering Document Version" />
  <IClassDef />
  <ISchemaObj DisplayName="Engineering document version" />
  <ISPFClassDefExt SPFIsConfigurationControlled="True"
SPFUniqueKeyDef="PARENT,SPFDocVersion" SPFFTREnabled="True" />
  </ClassDef>
- <Rel>
  <IObject UID="ModelClass.EngineeringDocVersion" />
  <IRel UID1="WorkBreakdownModel" UID2="EngineeringDocVersion" DefUID="ModelClass"
OrderValue="1" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="Instantiates.EngineeringDocVersion" />
  <IRel UID1="SchemaCompFactory" UID2="EngineeringDocVersion" DefUID="Instantiates"
OrderValue="1" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="Componentization.EngineeringDocVersion" />
  <IRel UID1="SPFAuthoringComponent" UID2="EngineeringDocVersion"
DefUID="Componentization" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
```

```
  </Rel>
- <InterfaceDef>
  <IObject UID="IEngineeringDocVersion" Name="IEngineeringDocVersion"
Description="Engineering Document Version" />
  <IInterfaceDef />
  <IInterfaceRole />
  <IPropCategory />
  <IEnumMetadata />
  <ISchemaObj DisplayName="Engineering document version" />
  </InterfaceDef>
- <Rel>
  <IObject UID="EngineeringDocVersion.IEngineeringDocVersion.PI" />
  <IRel UID1="EngineeringDocVersion" UID2="IEngineeringDocVersion"
DefUID="PrimaryInterface" OrderValue="1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="IEngineeringDocVersion.ISPFDesignDocVersion" />
  <IRel UID1="IEngineeringDocVersion" UID2="ISPFDesignDocVersion" DefUID="Implies"
OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.IObject" />
  <IRel UID1="EngineeringDocVersion" UID2="IObject" DefUID="Realizes" OrderValue="0"
IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFFileComposition" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFFileComposition" DefUID="Realizes"
OrderValue="10" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFOwnedObj" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFOwnedObj" DefUID="Realizes"
OrderValue="20" IsRequired="True" />
```

```
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFDocumentObject" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFDocumentObject" DefUID="Realizes"
OrderValue="30" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFDocumentVersion" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFDocumentVersion" DefUID="Realizes"
OrderValue="40" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFNonPublishedDocVersion" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFNonPublishedDocVersion"
DefUID="Realizes" OrderValue="50" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFExportableItem" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFExportableItem" DefUID="Realizes"
OrderValue="55" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFDesignDocVersion" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFDesignDocVersion" DefUID="Realizes"
OrderValue="60" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFSubscribableItem" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFSubscribableItem" DefUID="Realizes"
OrderValue="-1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
```

```
- <Rel>
  <IObject UID="EngineeringDocVersion.ISPFFTRItem" />
  <IRel UID1="EngineeringDocVersion" UID2="ISPFFTRItem" DefUID="Realizes"
OrderValue="-1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocVersion.IEngineeringDocVersion" />
  <IRel UID1="EngineeringDocVersion" UID2="IEngineeringDocVersion" DefUID="Realizes"
OrderValue="-1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocMaster.EngineeringDocRevision" />
  <IRel UID1="EngineeringDocMaster" UID2="EngineeringDocRevision"
DefUID="SPFIncludesClass" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocRevision.EngineeringDocVersion" />
  <IRel UID1="EngineeringDocRevision" UID2="EngineeringDocVersion"
DefUID="SPFIncludesClass" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
  </Container>
```

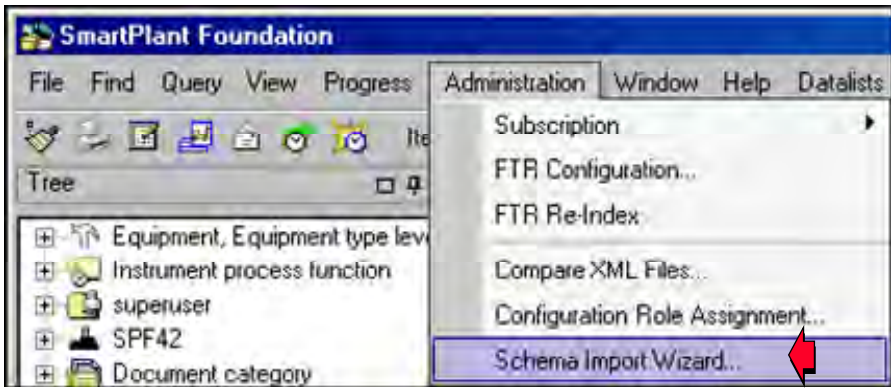
6.1.1 Importing the Model Additions

The modeled xml file will need to be loaded into the SPF database. Use the **Schema Import Wizard** from the Desktop Client to accomplish this task.

INTERGRAPH

Importing Schema Additions

- **In the SmartPlant Foundation Desktop Client, choose *Administration > Schema Import Wizard*.**

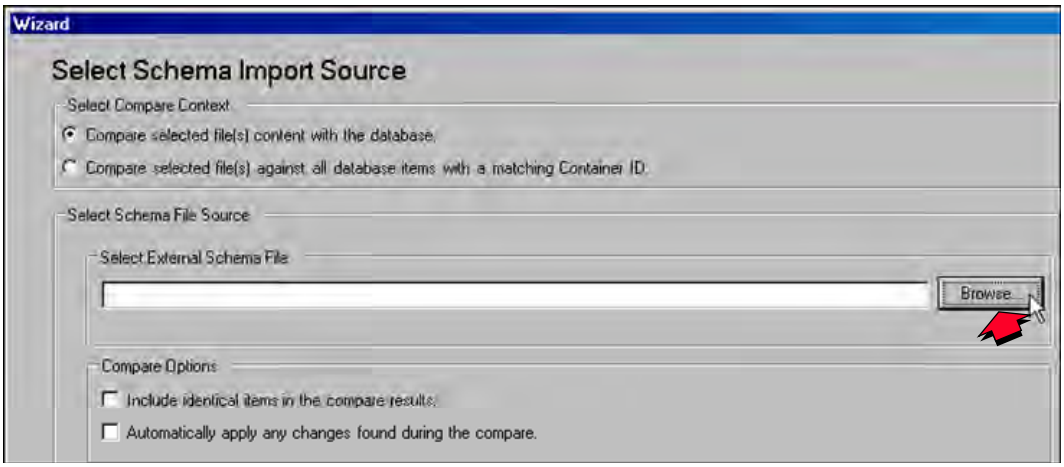


© 2008. Intergraph Corp.
All Rights Reserved.

The import *Wizard* form will appear.

Importing Schema Additions

- **In the *Schema Import Wizard*, click the *Browse* button to search for the schema file to be loaded.**

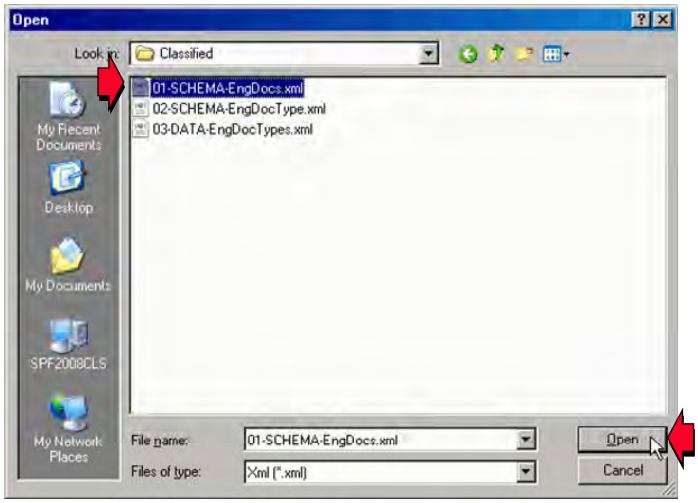


© 2008. Intergraph Corp.
All Rights Reserved.

When the browser dialog appears, select the schema file to be imported.

Importing Schema Additions

- **Select the *01-SCHEMA-EngDocs.xml* file and click Open.**

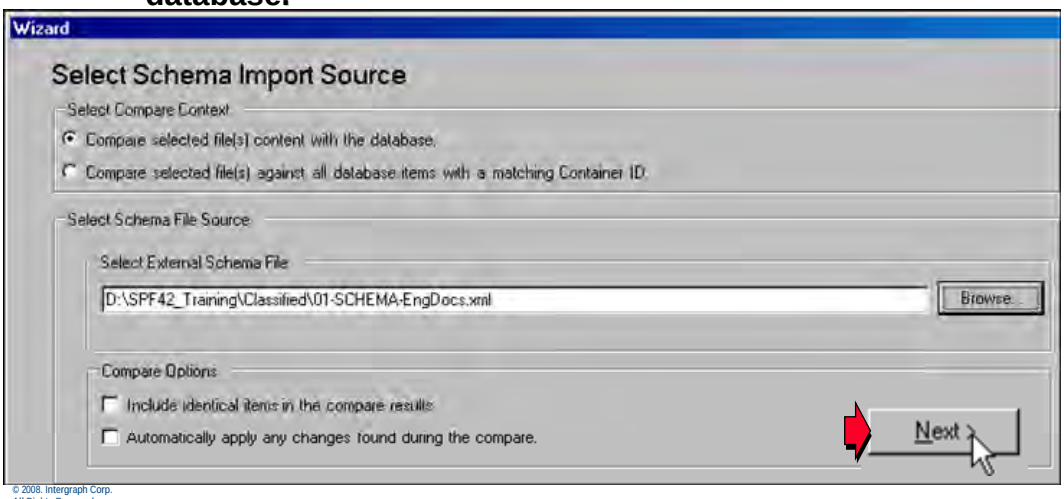


© 2008. Intergraph Corp.
All Rights Reserved.

When importing a schema into SmartPlant Foundation, it is necessary to first compare the schema with the existing content of the database and then selectively load the updates.

Importing Schema Additions

- **Click Next to compare schema file content with the database.**

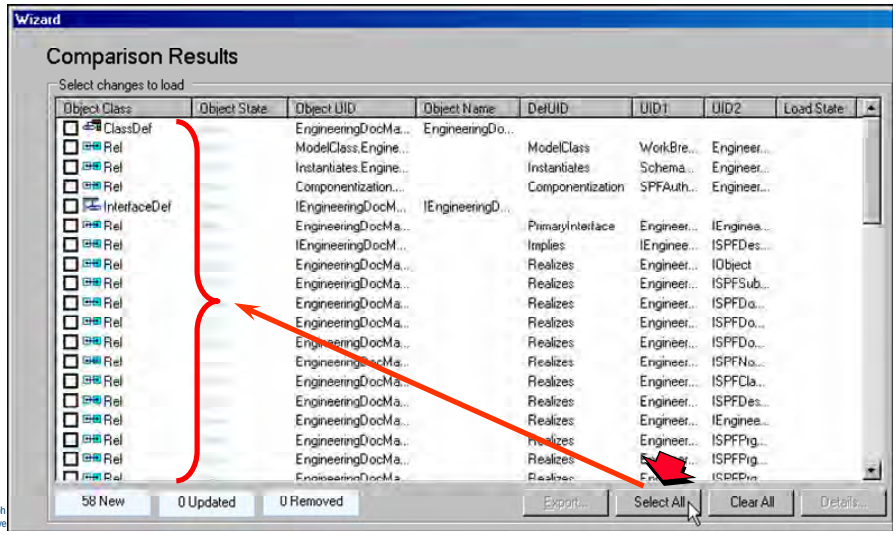


© 2008. Intergraph Corp.
All Rights Reserved.

The *Comparison Results* form will appear. To selectively load updates, enable the corresponding checkboxes next to the object to be loaded.

Importing Schema Additions

- **Choose the Select All button to import all definitions.**

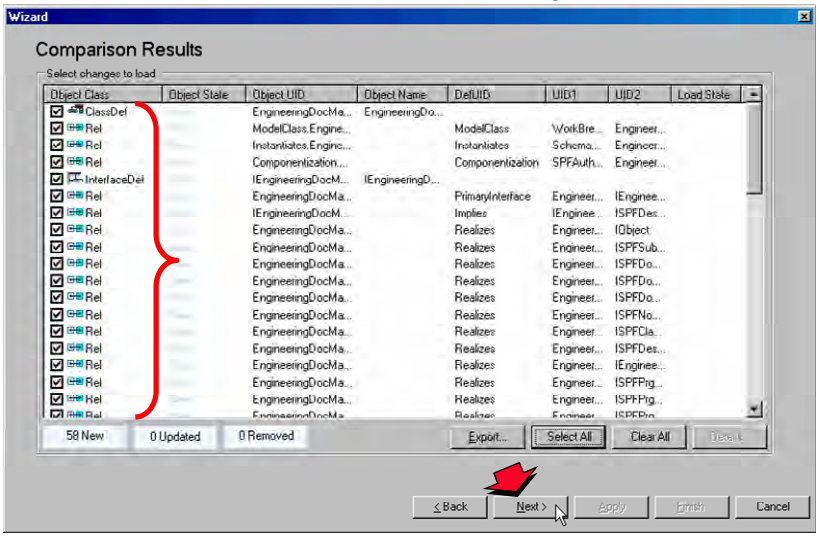


© 2008. Intergraph
All Rights Reserve

The totals (*New*, *Updated* and *Removed* objects and relationships) are displayed at the bottom of the form.

Importing Schema Additions

- **Click the Next button to start the import.**

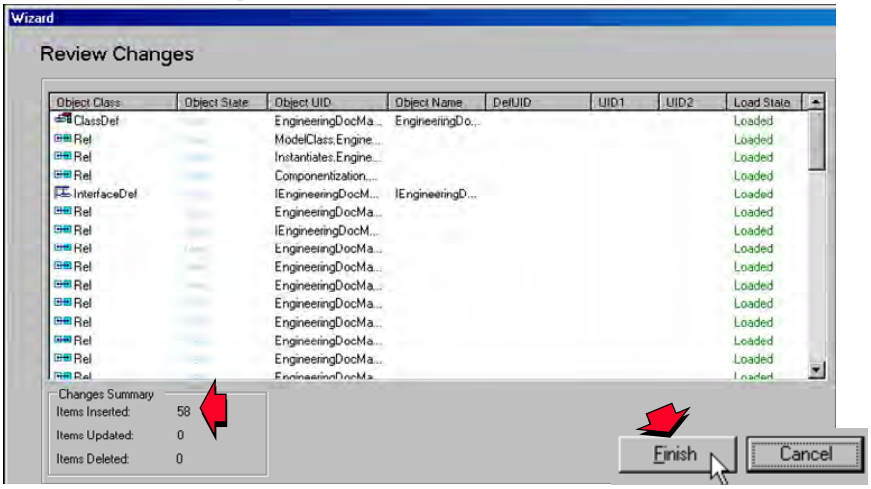


© 2008. Intergraph Corp.
All Rights Reserved.

After the update, the wizard displays the load status for each object and a summary in the *Review Changes* form.

Importing Schema Additions

- **Verify that the new objects are loaded to the database and click Finish.**



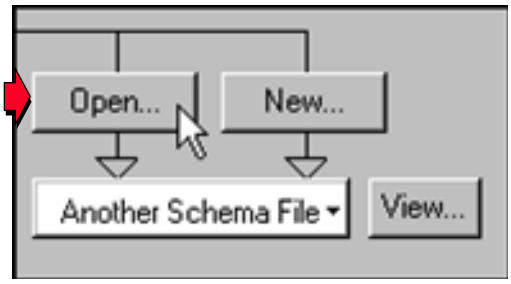
© 2008. Intergraph Corp.
All Rights Reserved.

6.1.2 Modeling a Document Classification Object

Use the *Schema Editor* to model a document classification object, called **EngineeringDocClass** with a primary interface of **IEngineeringDocClass**.

Modeling Classified Objects

- **Open the schema file containing the *Engineering Doc Types* document classifications by clicking the Open button above *Another Schema File.***



© 2008. Intergraph Corp.
All Rights Reserved.

The extension schema for this example is located in the *D:\SPF42_Training\Classified* folder.

Modeling Classified Objects

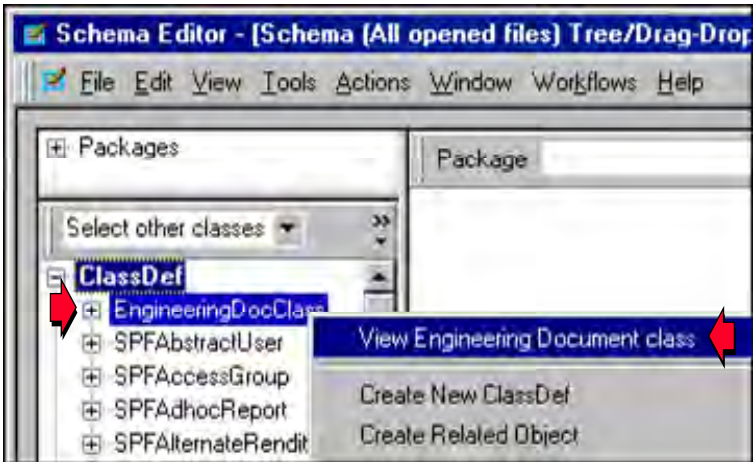
- **Select the *02-SCHEMA-EngDocType.xml* file and click Open.**

© 2008. Intergraph Corp.
All Rights Reserved.

Use the *View* command to see the structure of the *EngineeringDocClass* Class Def.

Modeling Classified Objects

- **In the *ClassDef* tree, right click on *EngineeringDocClass* and from the shortcut menu, select the command to View the Engineering Document class.**

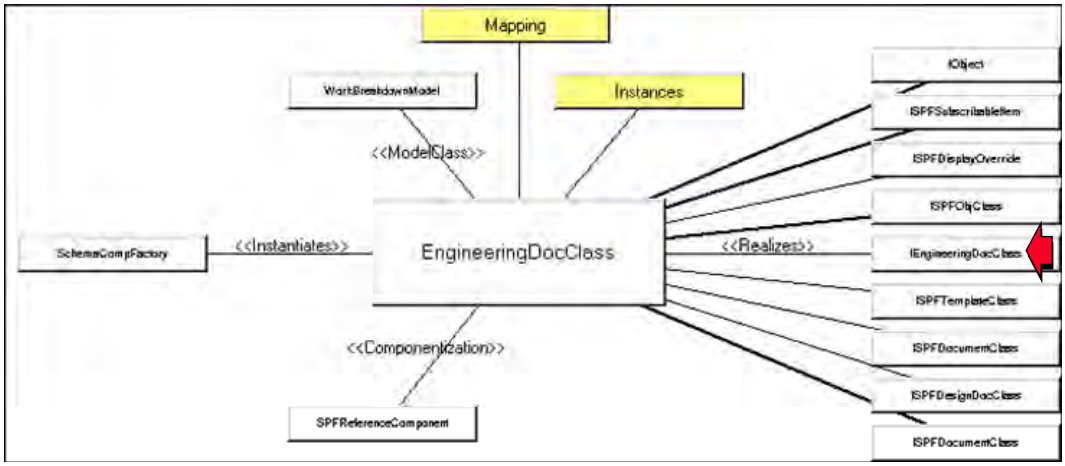


© 2008. Intergraph Corp.
All Rights Reserved.

This class has a primary interface called *IEngineeringDocClass*.

Modeling Classified Objects

This UML diagram shows the EngineeringDocClass document structure.

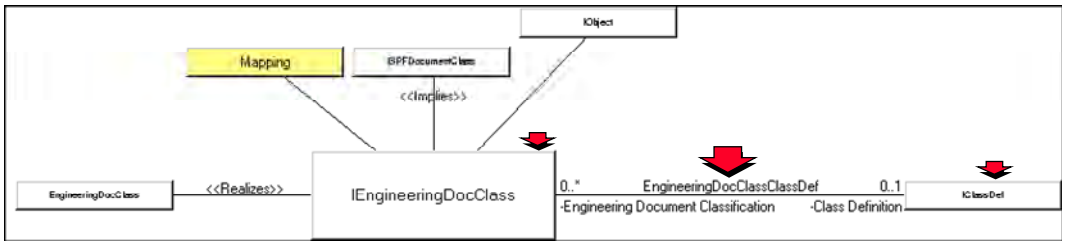


© 2008. Intergraph Corp.
All Rights Reserved.

In the Desktop Client when a new classification is created, a specialized RelDef will be needed.

Modeling Classified Objects

The EngineeringDocClassClassDef is a RelDef linking the *Engineering Document* class with the *ClassDef* (meta schema class).

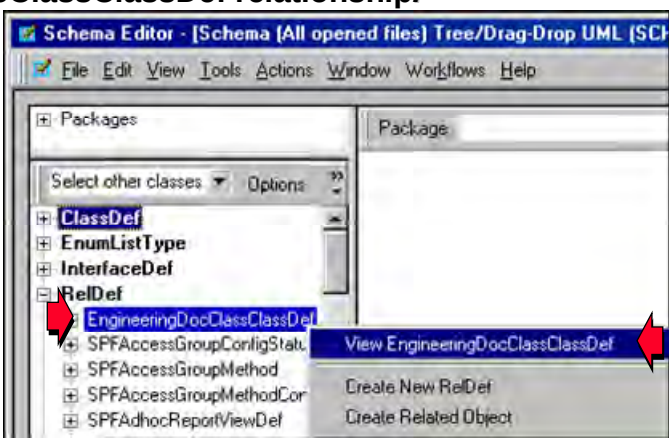


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *View* command to see the structure of the *EngineeringDocClassClassDef* RelDef.

Modeling Classified Objects

- **In the *RelDef* tree, right click on *EngineeringDocClascClassDef* and from the shortcut menu, select the command to View the Engineering DocClassClassDef relationship.**

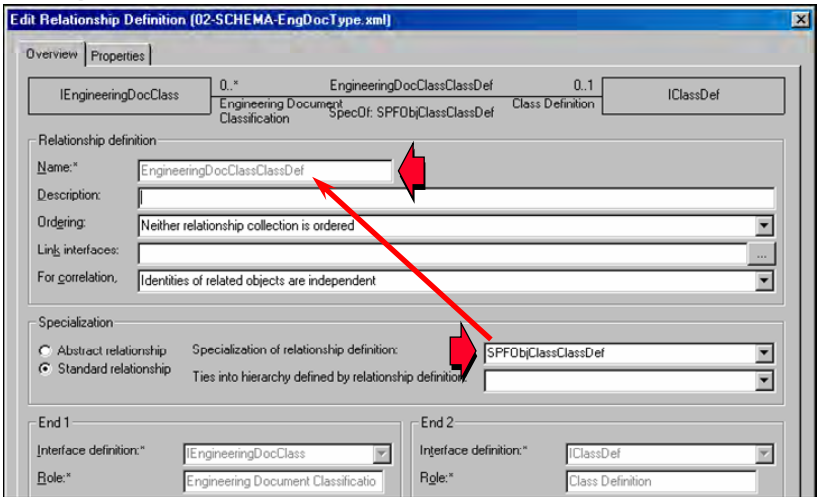


© 2008. Intergraph Corp.
All Rights Reserved.

In this example, it is called **EngineeringDocClassClassDef,** which is a specialization of *SPFObjClassClassDef*.

Modeling Classified Objects

EngineeringDocClassClassDef* which is a specialization of *SPFObjClassClassDef.



© 2008. Intergraph Corp.
All Rights Reserved.

Below is the **02-SCHEMA-EngDocType.xml** that creates the *class* and *relationship* for the **EngineeringDocClass** classified object.

```
- <Container CompSchema="MetaSchema" Scope="Schema" SoftwareVersion="04.02.00.04"
      IObject_ContainerID="Core.SPFDesignDocumentSchema" IObject_OBID="Auto"
      IObject_CreationDate="Now" IObject_TerminationDate="9999/12/31-23:59:59:999"
      IObject_LastUpdatedDate="Now" IObject_CreationUser="superuser"
      IObject_DomainUID="Auto" IObject_UniqueKey="Auto" IRel_DomainUID1="Auto"
      IRel_DomainUID2="Auto">
- <ClassDef>
  <IObject UID="EngineeringDocClass" Name="EngineeringDocClass"
      Description="Engineering Document class" />
  <IClassDef />
  <ISchemaObj DisplayName="Engineering Document class" />
  <ISPFClassDefExt SPFIsConfigurationControlled="False"
      SPFUIDDef="EDC,Name,SPFClassificationOccurance" />
  </ClassDef>
- <Rel>
  <IObject UID="Componentization.EngineeringDocClass" />
  <IRel UID1="SPFReferenceComponent" UID2="EngineeringDocClass"
      DefUID="Componentization" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="Instantiates.EngineeringDocClass" />
  <IRel UID1="SchemaCompFactory" UID2="EngineeringDocClass" DefUID="Instantiates"
      OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="ModelClass.EngineeringDocClass" />
  <IRel UID1="WorkBreakdownModel" UID2="EngineeringDocClass" DefUID="ModelClass"
      OrderValue="0" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <InterfaceDef>
  <IObject UID="IEngineeringDocClass" Name="IEngineeringDocClass"
      Description="Engineering Document class" />
  <IInterfaceDef />
  <IInterfaceRole />
  <IPropCategory />
  <IEnumMetadata />
  <ISchemaObj DisplayName="Engineering Document class" />
  </InterfaceDef>
- <Rel>
  <IObject UID="EngineeringDocClass.IEngineeringDocClass.PI" />
  <IRel UID1="EngineeringDocClass" UID2="IEngineeringDocClass"
      DefUID="PrimaryInterface" OrderValue="1" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="IEngineeringDocClass.ISPFDocumentClass" />
  <IRel UID1="IEngineeringDocClass" UID2="ISPFDocumentClass" DefUID="Implies"
      OrderValue="0" IsRequired="False" />
  <ISchemaObj />
```

```
  </Rel>
- <Rel>
  <IObject UID="IEngineeringDocClass.IObject" />
  <IRel UID1="IEngineeringDocClass" UID2="IObject" DefUID="Implies" OrderValue="0"
        IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.IEngineeringDocClass" />
  <IRel UID1="EngineeringDocClass" UID2="IEngineeringDocClass" DefUID="Realizes"
        OrderValue="10" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFDocumentClass" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFDocumentClass" DefUID="Realizes"
        OrderValue="20" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFDesignDocClass" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFDesignDocClass" DefUID="Realizes"
        OrderValue="30" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFTemplateClass" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFTemplateClass" DefUID="Realizes"
        OrderValue="20" IsRequired="False" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFDocumentClass" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFDocumentClass" DefUID="Realizes"
        OrderValue="50" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFObjClass" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFObjClass" DefUID="Realizes"
        OrderValue="10" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.IObject" />
  <IRel UID1="EngineeringDocClass" UID2="IObject" DefUID="Realizes" OrderValue="0"
        IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFDisplayOverride" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFDisplayOverride" DefUID="Realizes"
        OrderValue="5" IsRequired="False" />
  <ISchemaObj />
  </Rel>
```

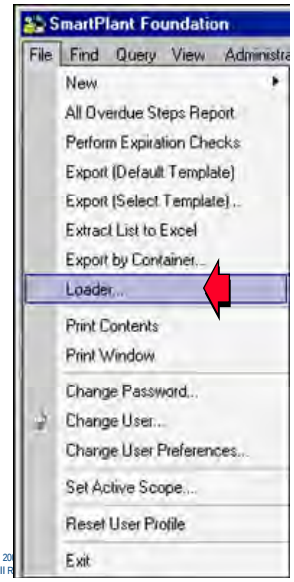
```
- <Rel>
  <IObject UID="EngineeringDocClass.ISPFSubscribableItem" />
  <IRel UID1="EngineeringDocClass" UID2="ISPFSubscribableItem" DefUID="Realizes"
        OrderValue="5" IsRequired="True" />
  <ISchemaObj />
  </Rel>
- <RelDef>
  <IObject UID="EngineeringDocClassClassDef" Name="EngineeringDocClassClassDef" />
  <IRel UID1="IEngineeringDocClass" UID2="IClassDef" />
  <IRelDef Role1="Engineering Document Classification" End1Locality="Across" Min1="0"
        Max1="*" SpecOfUID="SPFObjClassClassDef" Min2="0" Max2="1"
        End2Locality="Across" Role2="Class Definition" />
  <IRelDef2 />
  <ISchemaObj />
  <ISPFRelDefExt />
  </RelDef>
  </Container>
```

6.1.3 Loading and Viewing Classified Definitions

Use the **Loader** in the Desktop Client to load the *02-SCHEMA-EngDocType.xml* file (which will be shown later in this section).

INTERGRAPH

Loading Classified Objects

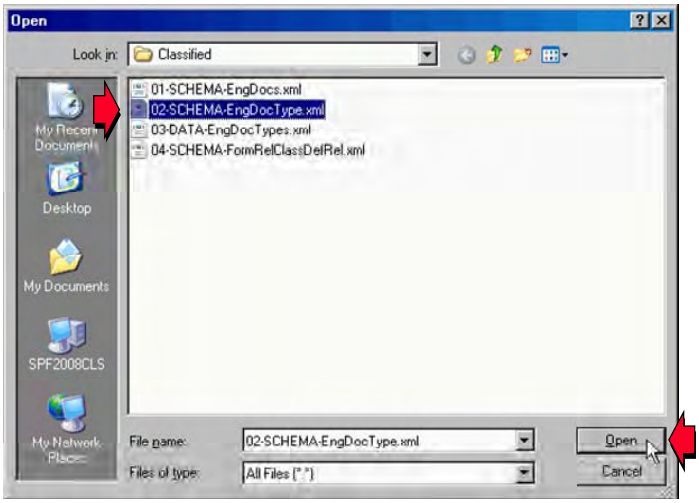


- **From the menu, click the *File > Loader* command.**

The *Open* dialog will appear.

Loading Classified Objects

- **Select the *02-SCHEMA-EngDocType.xml* file and click Open.**

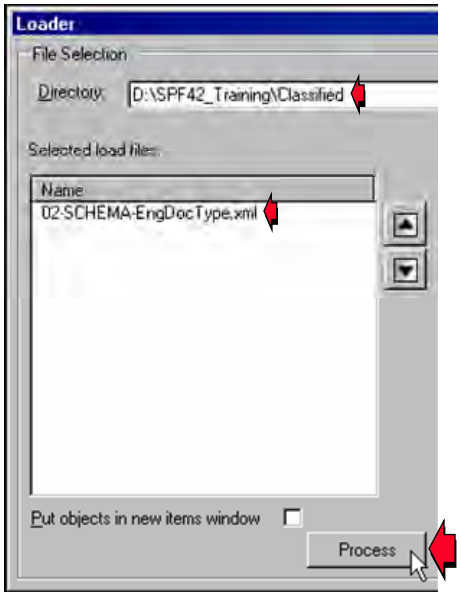


© 2008. Intergraph Corp.
All Rights Reserved.

The load file will display in the *Selected load files* field.

Loading Classified Objects

- **Click on the *Process* button to start the load.**



© 2008. Intergraph Corp.
All Rights Reserved.

Once the *EngineeringDocClass* has been loaded, use the ***Find*** > ***Schema*** > ***Class Definition*** menu command to display the *EngineeringDocClass* class definition.

INTERGRAPH

Viewing Classified Objects

- **From the Desktop Client, right click on the *EngineeringDocClass* ClassDef and from the shortcut menu, select *Expand ClassDef.***

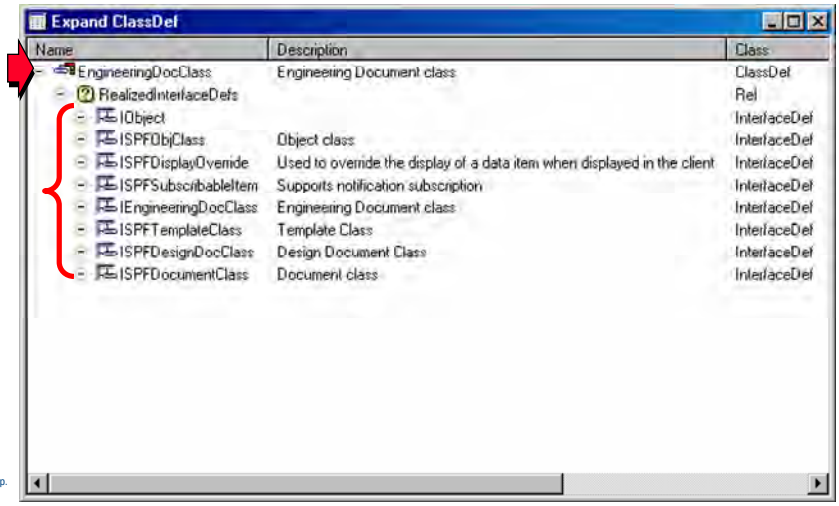
© 2008. Intergraph Corp.
All Rights Reserved.

The expanded ClassDef structure list view is displayed.

INTERGRAPH

Viewing Classified Objects

The list view will display the document classification object and its realized interfaces.

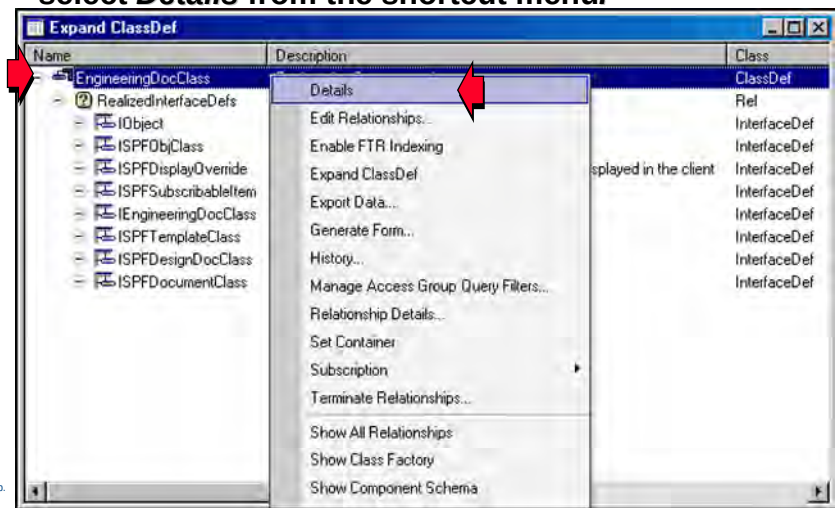


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *Details* command to view the specifics of the loaded class.

Viewing Classified Objects

- **Right click on the *EngineeringDocClass* ClassDef and select *Details* from the shortcut menu.**

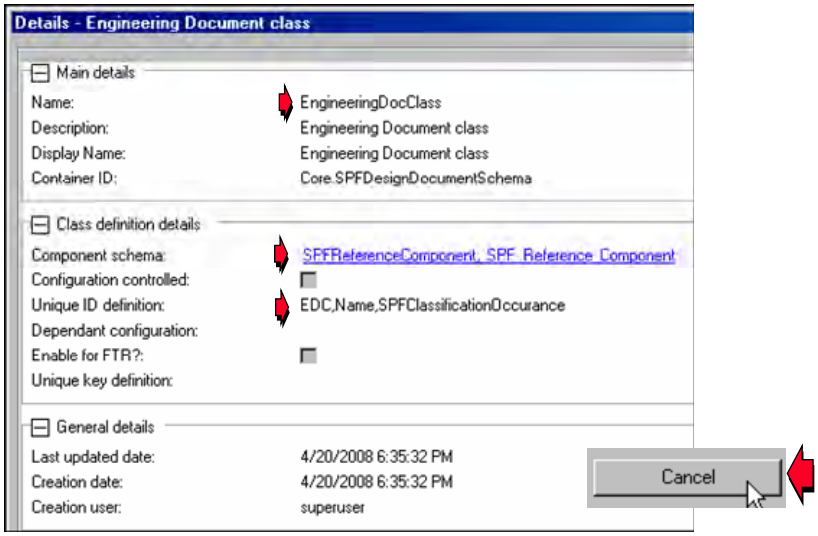


© 2008. Intergraph Corp.
All Rights Reserved.

The *Details* form will appear.

Viewing Classified Objects

- **Review the property values and click Cancel.**

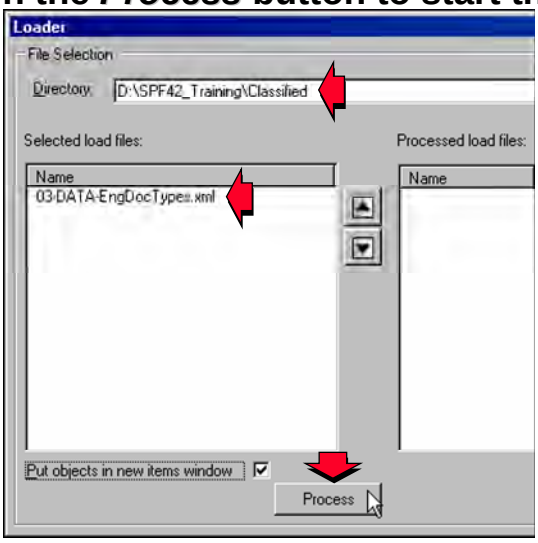


© 2008. Intergraph Corp.
All Rights Reserved.

The ***Loader*** command will also be used to process the *03-DATA-EngDocTypes.xml* file.

Loading Classified Objects

- **Click on the *Process* button to start the next load.**

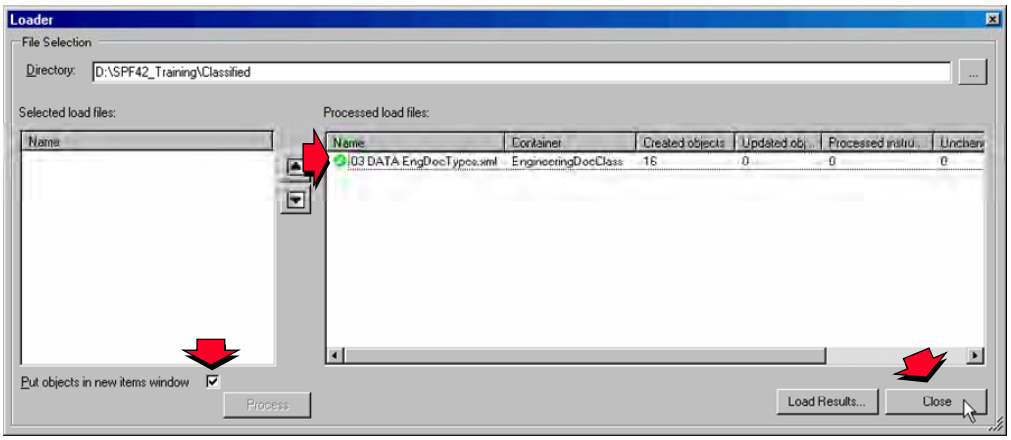


© 2008. Intergraph Corp.
All Rights Reserved.

By enabling the *Put objects in new items window*, the results of the load will be displayed.

Loading Classified Objects

- **Once the load files have processed successfully, click Close.**



© 2008. Intergraph Corp.
All Rights Reserved.

The **03-DATA-EngDocTypes.xml** load file is used to create a new engineering document classification tree like this:-

Engineering Doc Types

- Applied mechanics of materials
- Chemistry
- Communications
  - Wide area network
  - Satellite
- Composites
- Electronics

This file contains these classification objects and the relationships, but it also contains a default edge expansion which if it does not exist, then the classification tree will not appear from the ***File > New*** menu.

Since the 03-DATA-EngDocTypes.xml is a DATA file and not a SCHEMA file, it cannot be viewed in the Schema Editor.

Below is the **03-DATA-EngDocTypes.xml** that creates the *classification objects* and *relationships* for the **Engineering Doc Types**.

```
-IObject_TerminationDate="9999/12/31-23:59:59:999" IObject_UniqueKey="Auto"
      IObject_DomainUID="Auto" IObject_LastUpdatedDate="Now"
      IObject_CreationUser="superuser" IRel_DomainUID1="Auto"
      IRel_DomainUID2="Auto">
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Engineering_Document_Types_1" Name="Engineering Document
      Types" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
    </EngineeringDocClass>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Applied_mechanics_of_materials_1" Name="Applied mechanics of
      materials" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
    </EngineeringDocClass>
- <Rel>
```

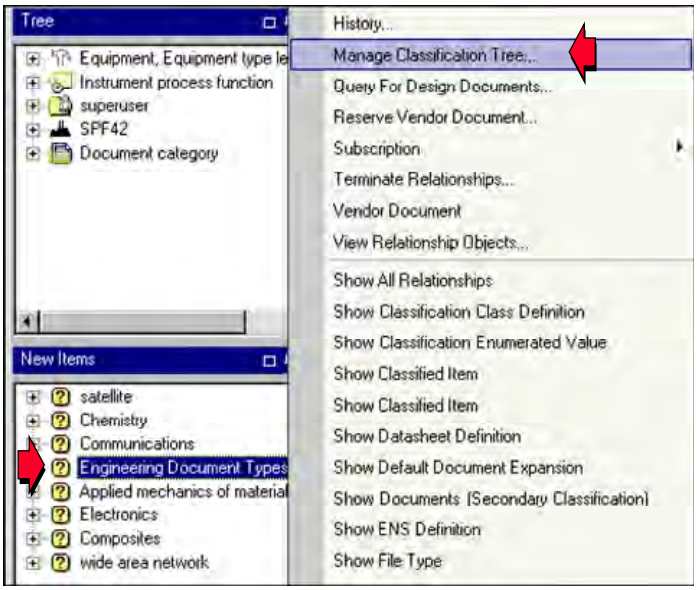
```
  <IObject
      UID="EDC_Engineering_Document_Types_1.EDC_Applied_mechanics_of_materials_1"
      />
  <IRel UID1="EDC_Engineering_Document_Types_1"
      UID2="EDC_Applied_mechanics_of_materials_1" DefUID="SPFClassMember"
      IsRequired="False" OrderValue="0" />
  <ISchemaObj />
    </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Chemistry_1" Name="Chemistry" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
    </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Engineering_Document_Types_1.EDC_Chemistry_1" />
  <IRel UID1="EDC_Engineering_Document_Types_1" UID2="EDC_Chemistry_1"
      DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
    </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Communications_1" Name="Communications" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
    </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Engineering_Document_Types_1.EDC_Communications_1" />
  <IRel UID1="EDC_Engineering_Document_Types_1" UID2="EDC_Communications_1"
      DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
    </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_wide_area_network_1" Name="wide area network" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
    </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Communications_1.EDC_wide_area_network_1" />
  <IRel UID1="EDC_Communications_1" UID2="EDC_wide_area_network_1"
      DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
    </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_satellite_1" Name="satellite" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
```

```
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
     </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Communications_1.EDC_satellite_1" />
  <IRel UID1="EDC_Communications_1" UID2="EDC_satellite_1"
        DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
     </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Composites_1" Name="Composites" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
     </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Engineering_Document_Types_1.EDC_Composites_1" />
  <IRel UID1="EDC_Engineering_Document_Types_1" UID2="EDC_Composites_1"
        DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
     </Rel>
- <EngineeringDocClass>
  <IEngineeringDocClass />
  <IObject UID="EDC_Electronics_1" Name="Electronics" Description="" />
  <ISPFObjClass SPFClassificationOccurance="1" />
  <ISPFSubscribableItem />
  <ISPFDocumentClass />
  <ISPFDesignDocClass />
     </EngineeringDocClass>
- <Rel>
  <IObject UID="EDC_Engineering_Document_Types_1.EDC_Electronics_1" />
  <IRel UID1="EDC_Engineering_Document_Types_1" UID2="EDC_Electronics_1"
        DefUID="SPFClassMember" IsRequired="False" OrderValue="0" />
  <ISchemaObj />
     </Rel>
- <Rel>
  <IObject UID="EngineeringDocClass.DefaultEdge" />
  <IRel UID1="EngineeringDocClass" UID2="EDG_DefaultDocExpansion"
        DefUID="SPFClassDefDefaultEdgeDef" OrderValue="0" IsRequired="False" />
  <ISchemaObj />
     </Rel>
     </Container>
```

6.2 Viewing the Classification Tree

Use the Desktop Client to view the definitions that have been loaded. Because the *Put objects in new items window* toggle was enabled, the loaded **Engineering Document Types** objects are visible in the *New Items* window.

Managing Classifications

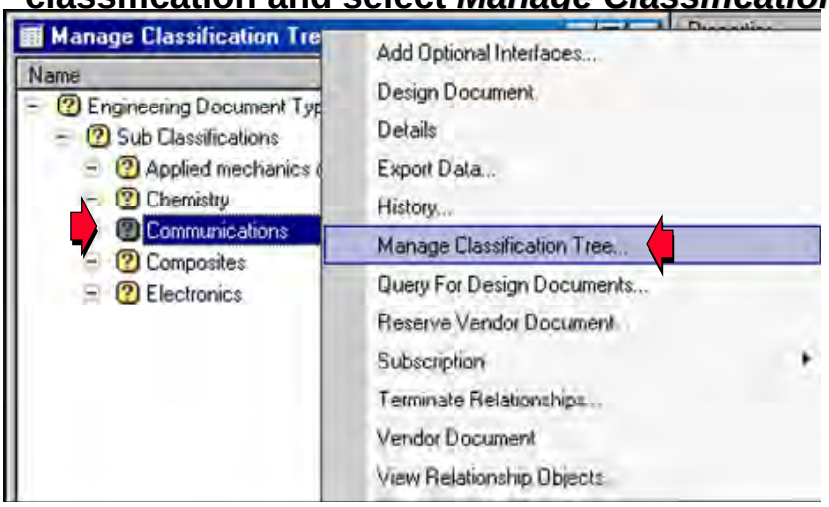


- **Right click on the *Engineering Document Types* classification and select *Manage Classification Tree* from the shortcut menu.**

The *Manage Classification Tree* list view will appear.

Managing Classifications

- **Right click on the *Communications* sub classification and select *Manage Classification***

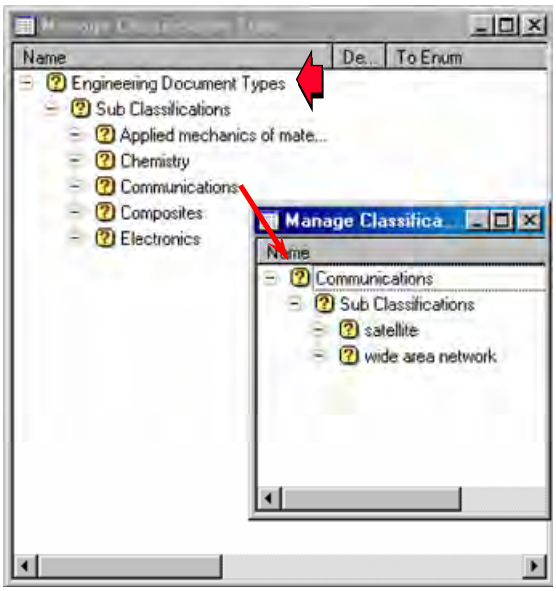


© 2008. Intergraph Corp.
All Rights Reserved.

Another *Manage Classification Tree* list view will appear to show sub classifications.

Managing Classifications

This shows the Engineering Document Types classifications and sub classifications.

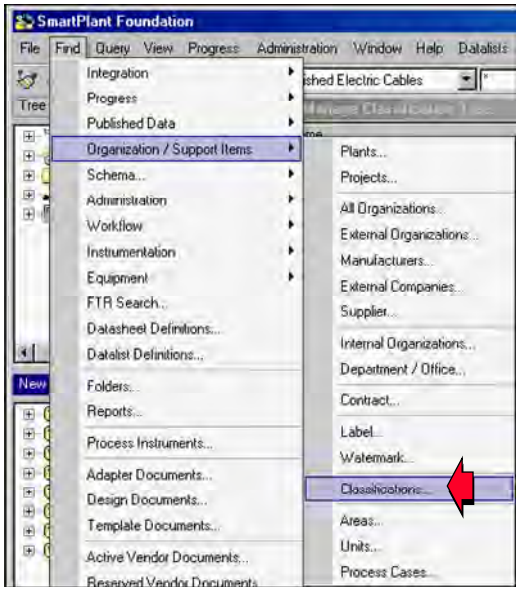


© 2008. Intergraph Corp.
All Rights Reserved.

If the Classification objects are not visible in the *New Items* window, a search can be performed from the menu.

Managing Classifications

- **From the menu, click *Find > Organization / Support Items > Classifications*.**

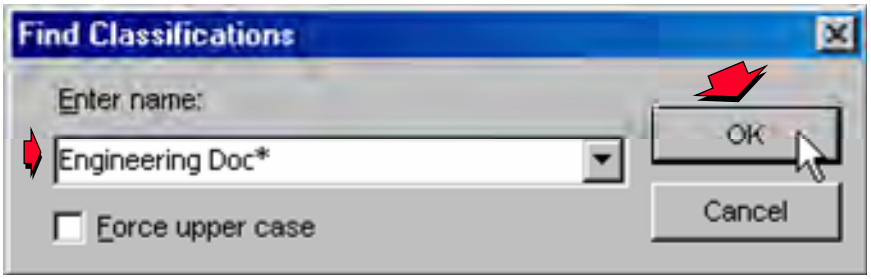


© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Classifications* dialog will appear.

Managing Classifications

- **Enter the criteria to find and display all *Classifications* that start with Engineering Doc, then click OK.**

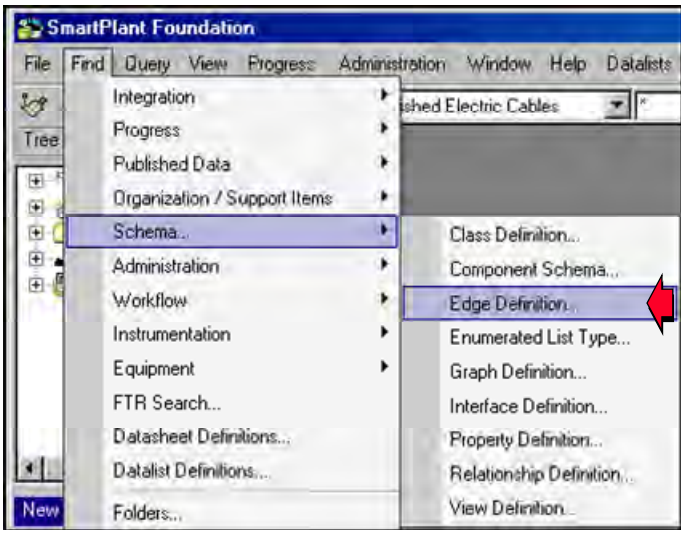


© 2008. Intergraph Corp.
All Rights Reserved.

Use the ***Find*** command to view the *Edge Definition* that was loaded.

Managing Classifications

- **From the menu, click *Find* > *Schema* > *Edge Definition*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Edge Definition* dialog will appear.

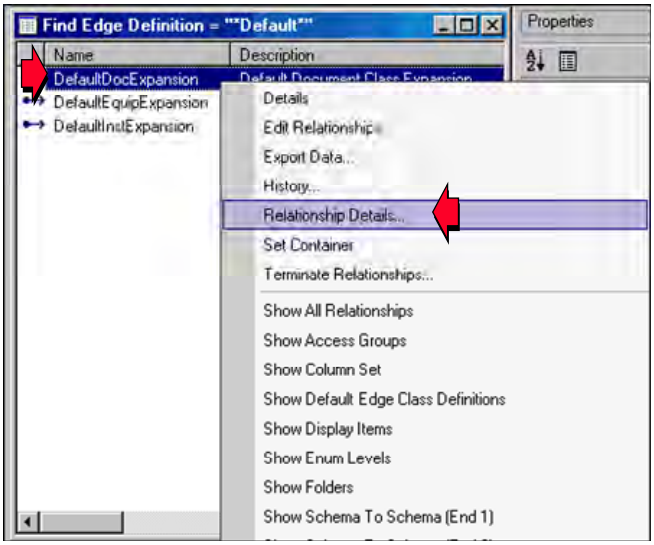
Managing Classifications

- **Enter the criteria to find and display all *Edge Defs* that have *Default* in the name and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

The results of the find will appear in a list view window.

Managing Classifications



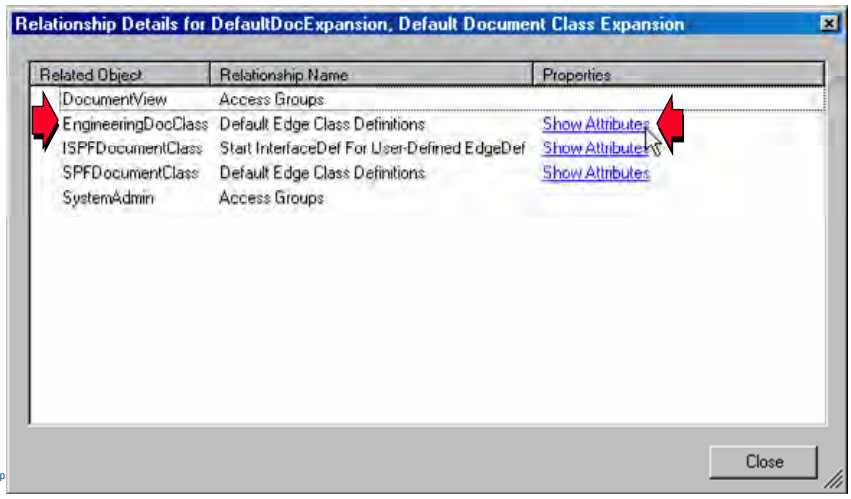
- **Right click on the *DefaultDocExpansion* Edge Definition and select *Relationship Details* from the shortcut menu.**

© 2008. Intergraph Corp.
All Rights Reserved.

A *Relationship Details* form will appear.

Managing Classifications

- **Click on the *Show Attributes* hyperlink for the *EngineeringDocClass* related object.**



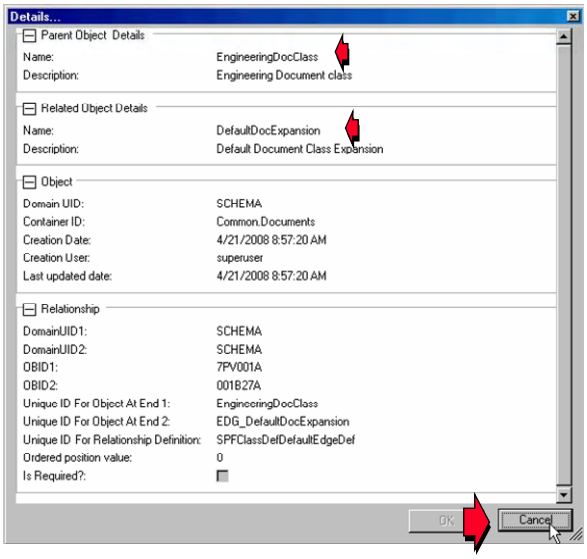
© 2008. Intergraph Corp.
All Rights Reserved.

A *Details* form will appear showing the EdgeDef that was loaded.

INTERGRAPH

Managing Classifications

- **View the Details for the EngineeringDocClass default expansion relationship.**
- **Click Cancel to close the form.**



© 2008. Intergraph Corp.
All Rights Reserved.

6.3 Creating a New Form for Classification

When setting up a Classification Tree, the new classified object, *Engineering Document*, could be related to an existing form, *Design Document*. However, when using heterogeneous classifications, it is not clear that the new document (Engineering Document) belongs to a different ClassDef.

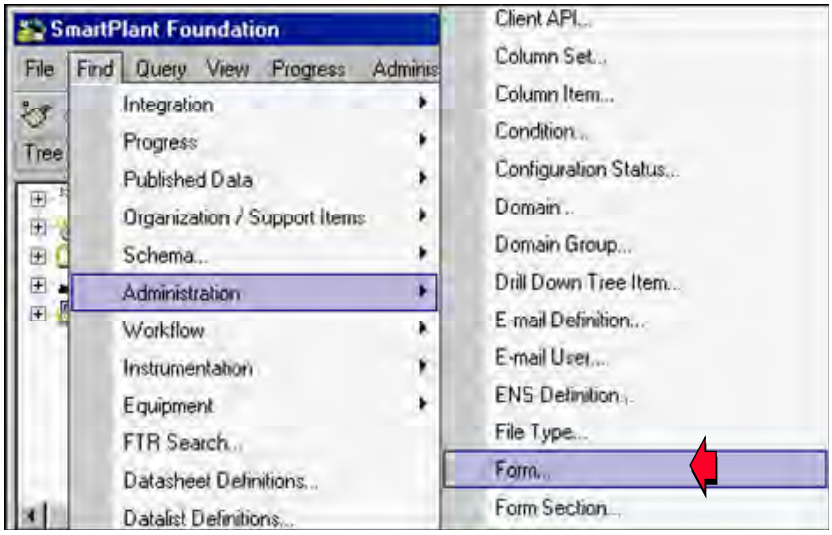
To see the differences in a heterogeneous classification, a copy of the Design Document form will be made and the new copy related to the EngineeringDocVersion ClassDef. In addition, the Engineering Numbering System (ENS) will be used to set a new name for any instances of the Engineering Document.

Start by finding the existing Design Document form to use for making the copy.

INTERGRAPH

Copying a Form

- ❑ **Select *Find > Administration > Form* from the**

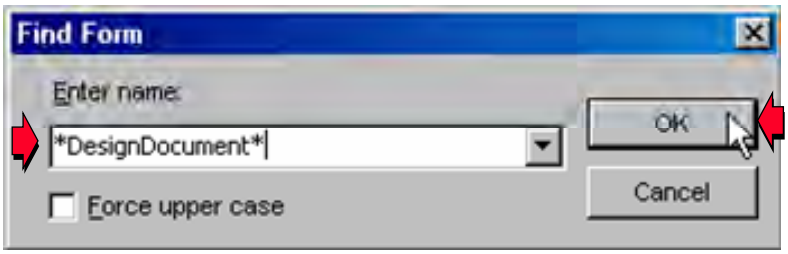


© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Form* dialog appears.

Copying a Form

- **Enter the criteria to find the existing *DesignDocument* form and click OK.**

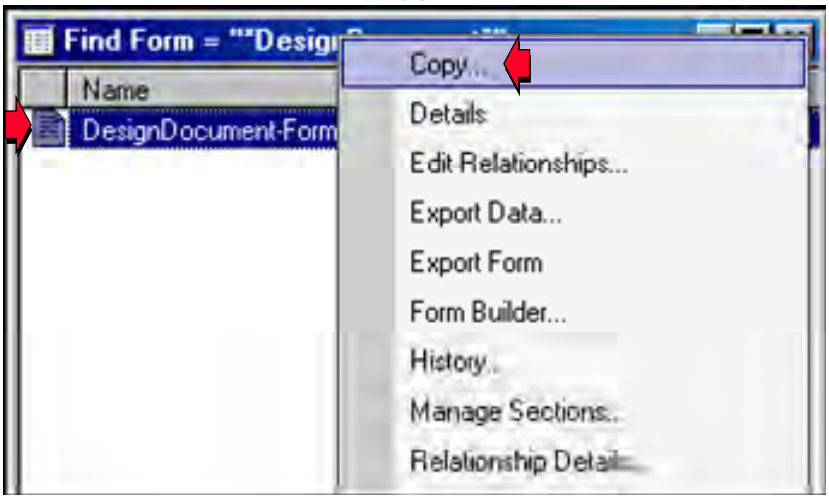


© 2008. Intergraph Corp.
All Rights Reserved.

From the list view window, select the form to be copied.

Copying a Form

- **Right click on the *DesignDocument-Form* and from the shortcut menu, select *Copy*.**

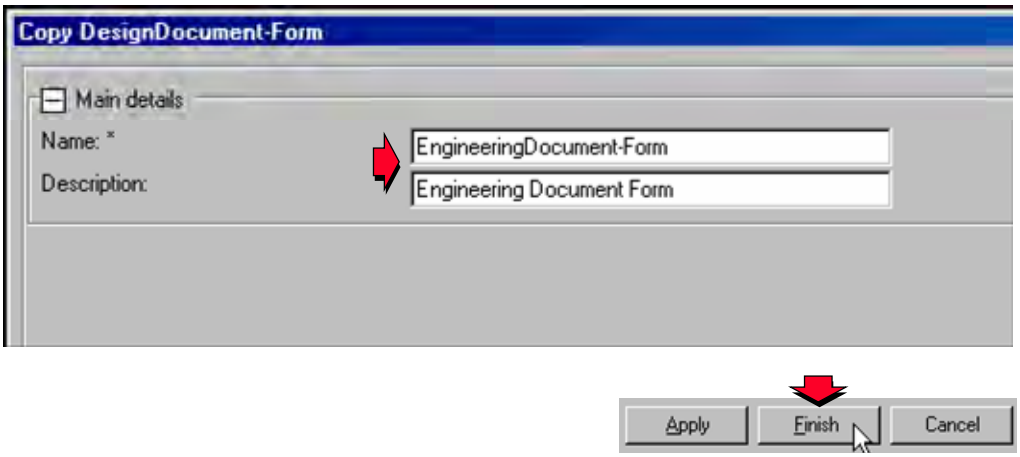


© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Form* form will appear.

Copying a Form

- **Enter a new required *Name* and optional *Description* for the form copy and click Finish.**

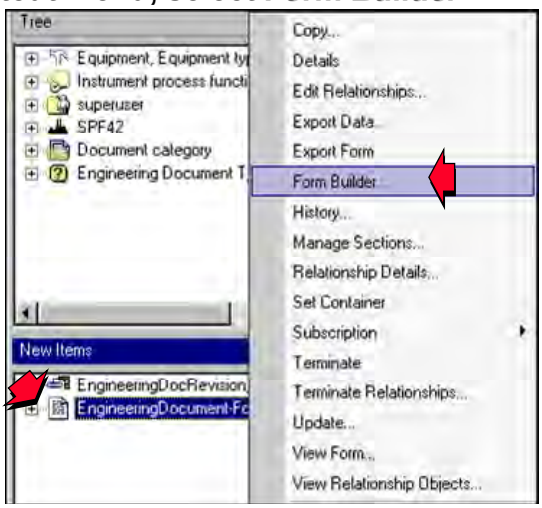


© 2008. Intergraph Corp.
All Rights Reserved.

Use **Form Builder** to edit the form sections and display items.

Copying a Form

- **Right click on the *EngineeringDocument-Form* and from the shortcut menu, select *Form Builder*.**

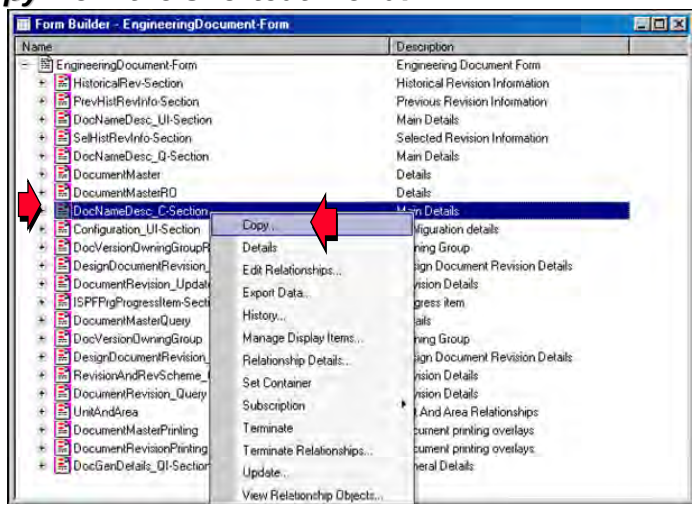


© 2008. Intergraph Corp.
All Rights Reserved.

The *Form Builder* window will show the **form**, related **sections** and related **display items**.

Copy Section

- **Right click on the *DocNameDesc_C* section and select *Copy* from the shortcut menu.**

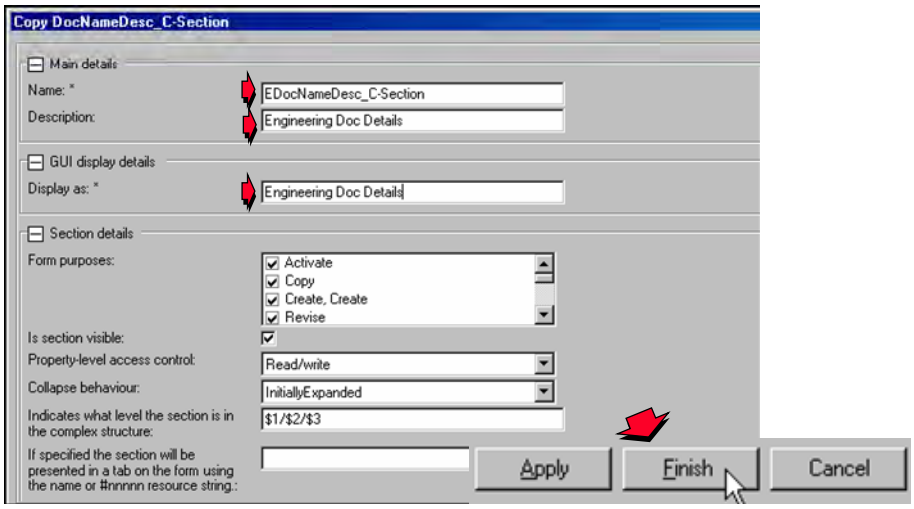


© 2008. Intergraph Corp.
All Rights Reserved.

Copying a section allows you to create a new section definition that references the same display items as the original section. It also copies all the associated relationships.

Copy Section

- **Change the appropriate fields to define a new *Form Section* based on the selected section.**

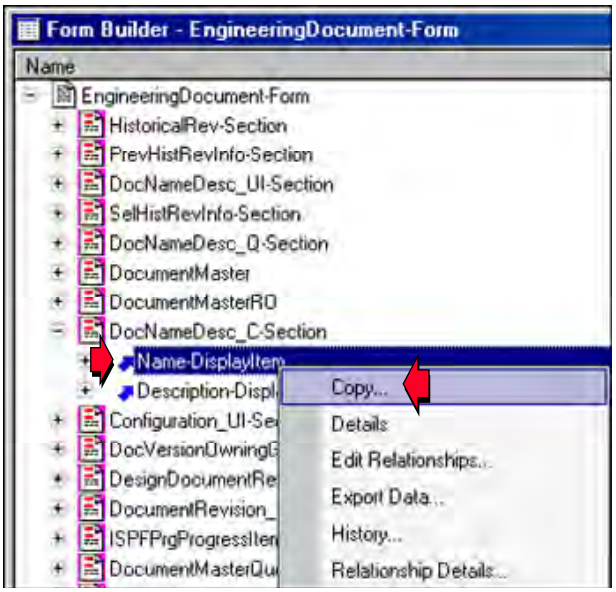


© 2008. Intergraph Corp.
All Rights Reserved.

When dealing with complex objects a section (e.g. Main details) could exist for each of the complex objects. The $1/$2/$3 indicate which one of the complex objects this section is used for. Make a copy of the existing **Name** display item so that any changes will only affect the *EngineeringDocument* form.

Copy Display Items

- **Right click on *Name-DisplayItem* and from the shortcut menu, select *Copy*.**

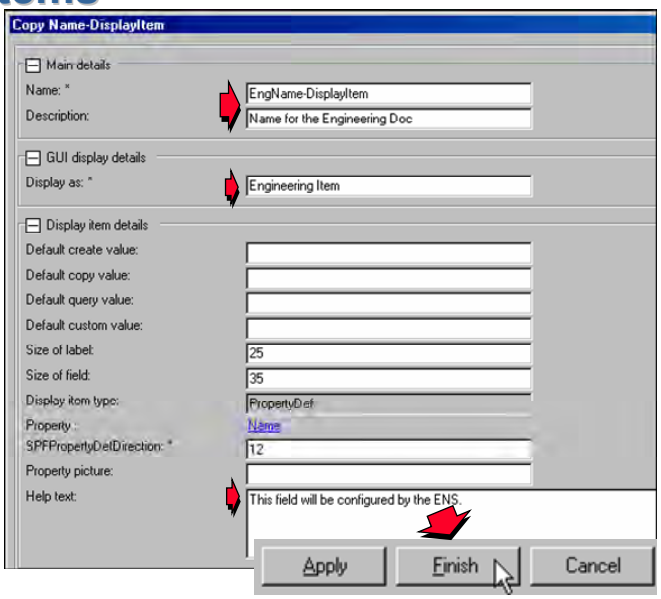


© 2008. Intergraph Corp.
All Rights Reserved.

The *Copy Name DisplayItem* form will appear.

Copy Display Items

- **Change the appropriate values to define a new *Display Item* based on the existing item and click Finish.**

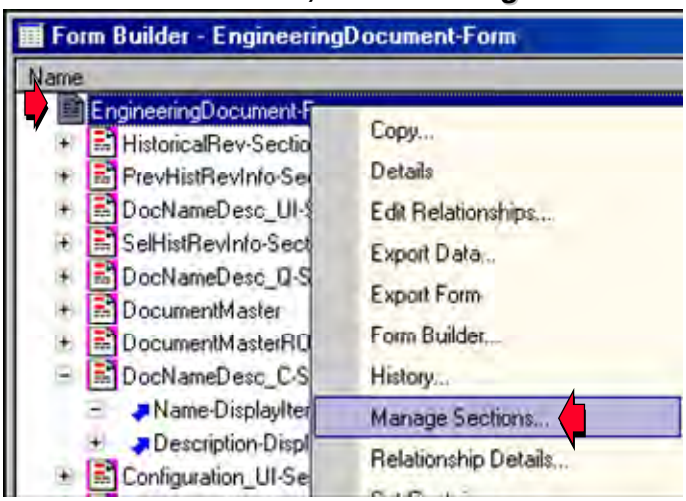


© 2008. Intergraph Corp.
All Rights Reserved.

A form's sections can be managed through the *Manage Sections* Form. This form can be used to *create*, *update* and *re-order* sections, *terminate* relationships, and *delete* sections.

Manage Form Sections

- **Right click on the *EngineeringDocument-Form* object and from the shortcut menu, select *Manage Sections*.**

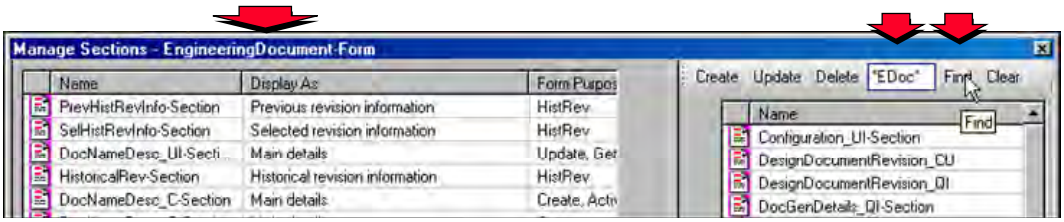


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Sections* form will appear**.**

Manage Form Sections

- **To find the new section, enter **EDoc** in the search field and click Find.**



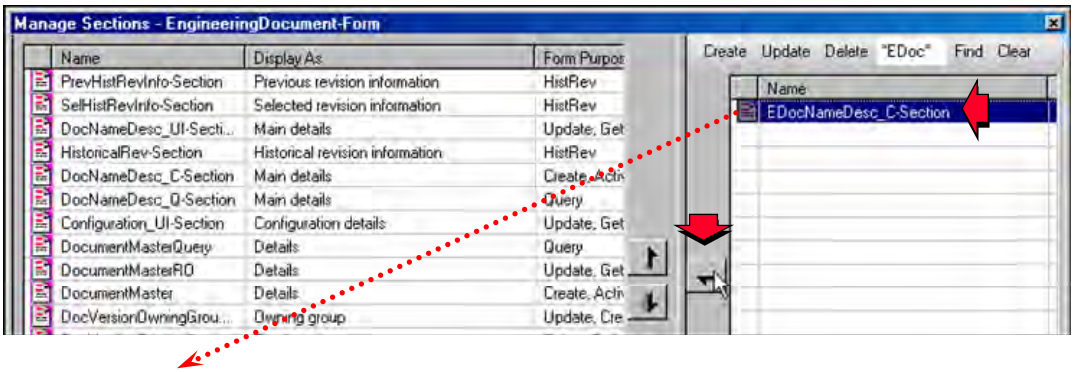
© 2008. Intergraph Corp.
All Rights Reserved.

The results are displayed in the **Library** pane (on the right).

Add the new section to the *EngineeringDocument* form

Manage Form Sections

- **Select the *EDocNameDesc_C-Section* and click the left arrow button to relate the item to the copied EngineeringDocument-Form.**

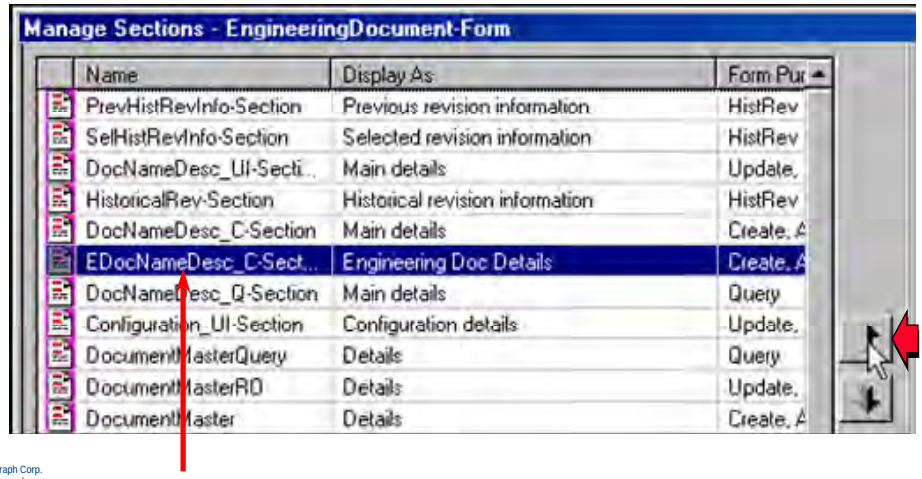


© 2008. Intergraph Corp.
All Rights Reserved.

Select the new section and use the arrows to change the section order.

Manage Form Sections

- **Use the up arrow to move the new section below the original *DocNameDesc_C-Section*.**

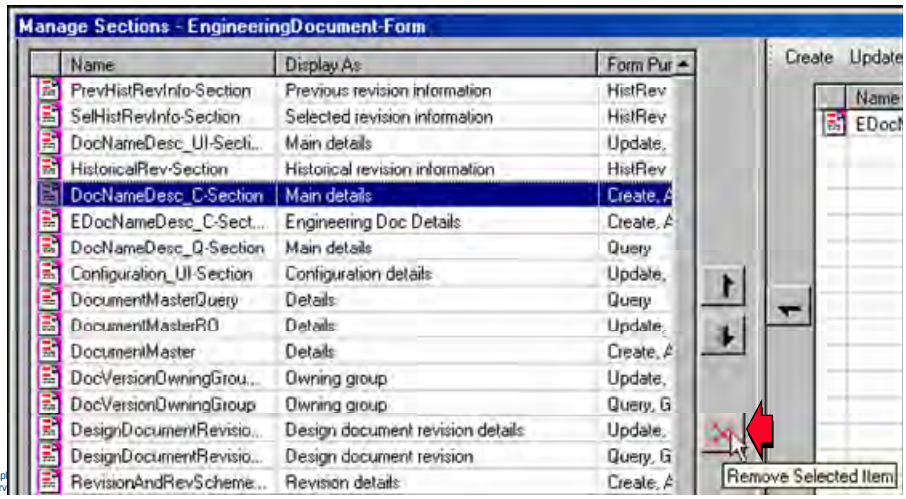


© 2008. Intergraph Corp.
All Rights Reserved.

To delete a section from a form, it is only necessary to delete the relationship.

Manage Form Sections

- **Select the original *DocNameDesc_C-Section* and click the delete button to remove the section from the form.**

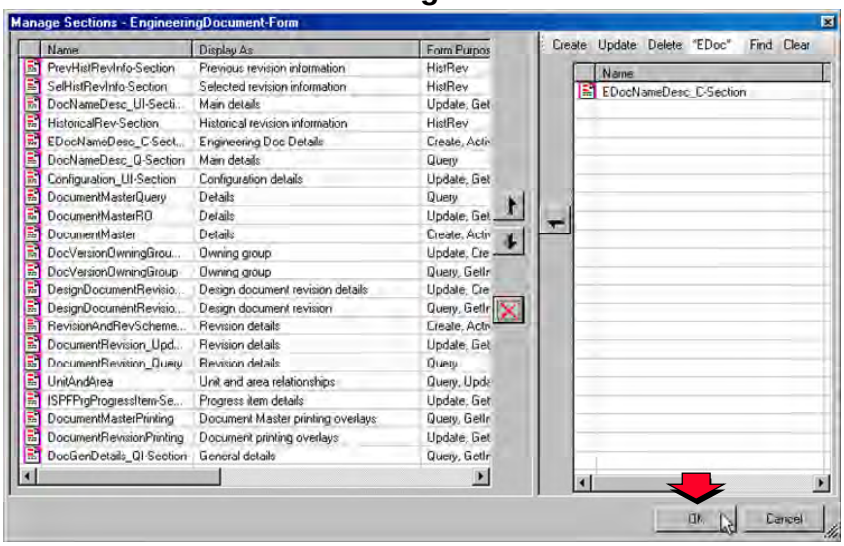


© 2008. Intergraph
All Rights Reserv

The section will remain in the library for use by other forms.

Manage Form Sections

- **Click OK to save the changes to the form.**

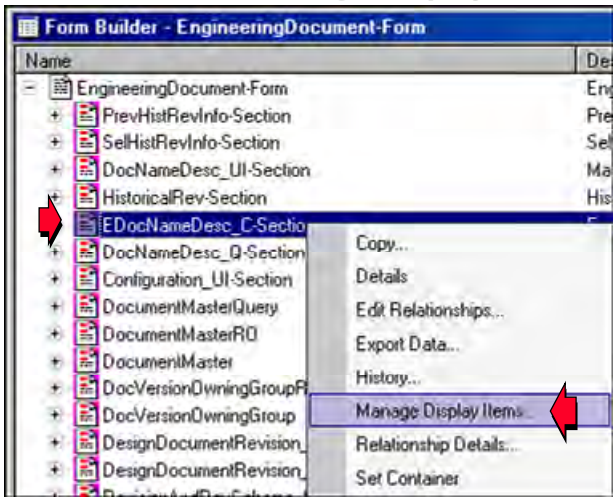


© 2008. Intergraph Corp.
All Rights Reserved.

The display items in a section can be managed through the *Manage Display Items* form. This form can be used to *create*, *update* and *re-order* display items, *terminate* relationships, and *delete* display items.

Manage Display Items

- **Right click *EDocNameDesc_C-Section* and from the shortcut menu, select *Manage Display Items.***

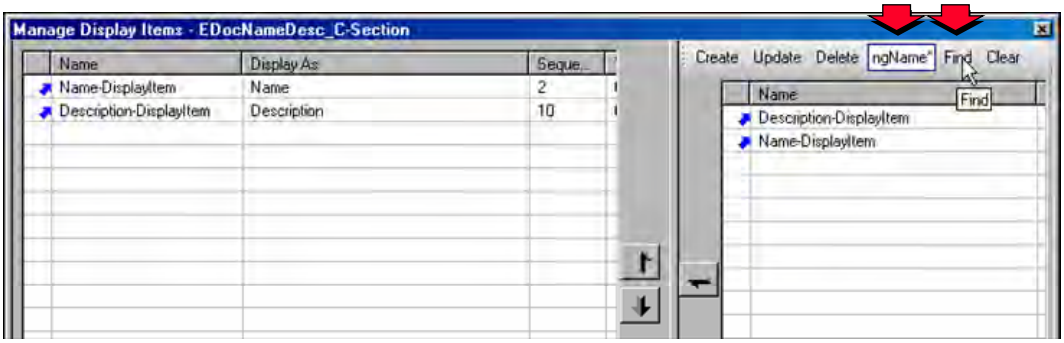


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Display Items* form will appear.

Manage Display Items

- **To find the new display item, enter **EngName** in the search field and click Find.**

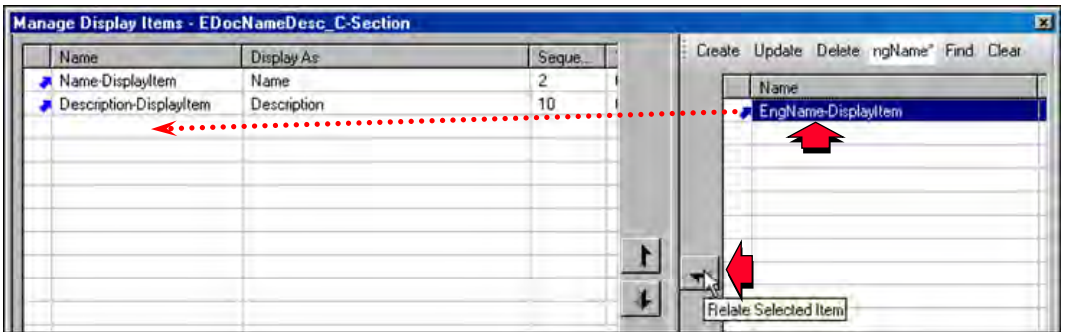


© 2008. Intergraph Corp.
All Rights Reserved.

The results are displayed in the *Library* pane.

Manage Display Items

- **Select *EngName-DisplayItem* and click the left arrow button to relate the item to the *EDocNameDesc_C-Section* section.**

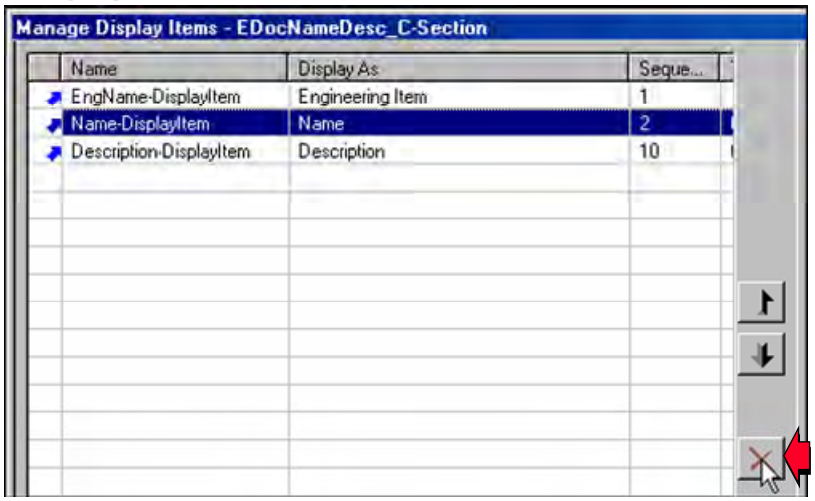


© 2008. Intergraph Corp.
All Rights Reserved.

Change the *DisplayAs* value for the display item to reflect the name as **Engineering Item** as shown below.

Manage Display Items

- **Use the delete button, to delete the original *Name-DisplayItem*.**

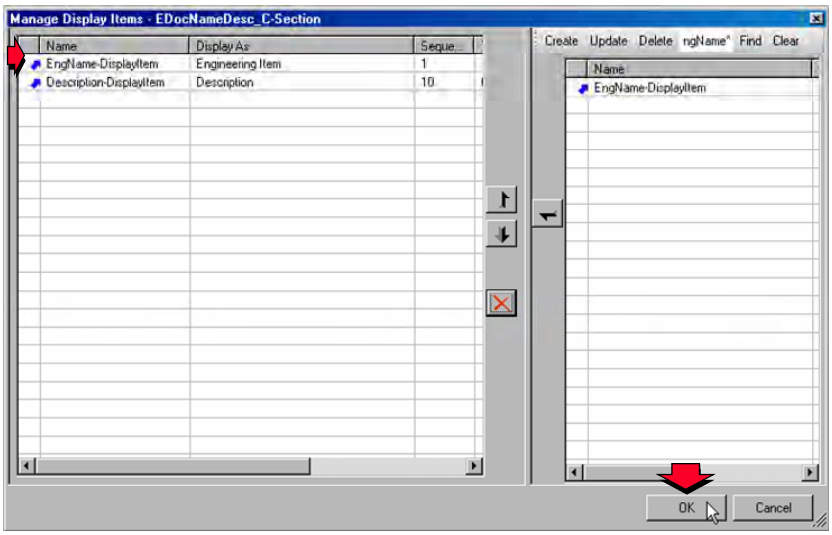


© 2008. Intergraph Corp.
All Rights Reserved.

To delete a display item from a section, it is only necessary to delete the relationship. The display item will remain in the library for use by other sections.

Manage Display Items

- **To save the section changes, click OK.**

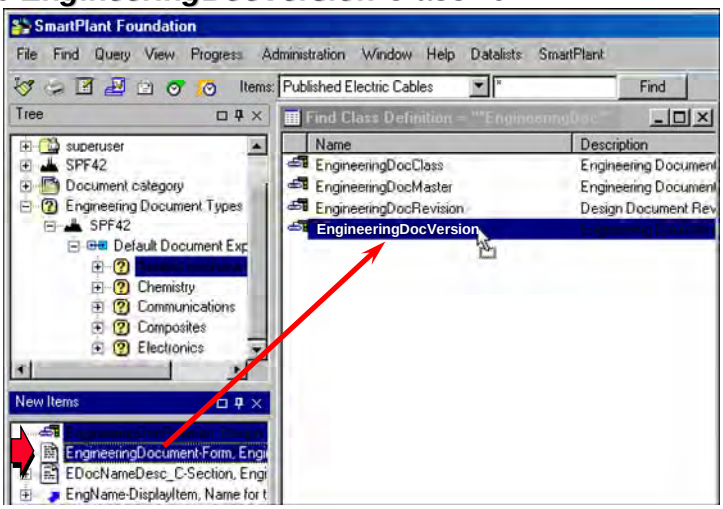


© 2008. Intergraph Corp.
All Rights Reserved.

Finally, save the changes in the *Manage Display Items* form. Next, relate the new form to the **EngineeringDocVersion** class definition using drag and drop

Relating a New Form

- **Drag and drop the copied *EngineeringDocument-Form* onto the *EngineeringDocVersion* ClassDef.**

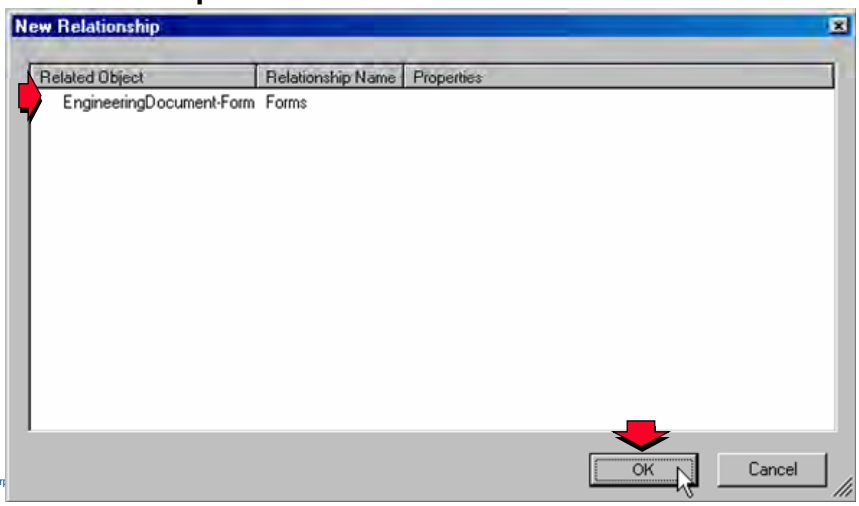


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Relationship* form will appear.

Relating a New Form

- **Click OK to create the new Forms (Form → ClassDef) relationship.**



© 2008. Intergraph Corp
All Rights Reserved.

6.3.1 Loading the Form Relationship

Next, define a relationship between the ClassDef and the form so the form is visible in the GUI. The relationship tells the server what ClassDef to create from the classification.

This is contained in the load file **04-SCHEMA-FormRelClassDefRel.xml.** This shows a single homogeneous classification tree creating just one ClassDef. For a heterogeneous classification, create another *EngineeringDocClassClassDef* relationship from a lower down classification like Communications to say Design document. This will be shown later in this chapter.

Below is the **04-SCHEMA-FormRelClassDefRel.xml** that creates these *relationships*.

```
- <Container CompSchema="MetaSchema" Scope="Schema" SoftwareVersion="04.02.00.04"
      IObject_ContainerID="Core.SPFDesignDocumentSchema" IObject_OBID="Auto"
      IObject_CreationDate="Now" IObject_TerminationDate="9999/12/31-23:59:59:999"
      IObject_LastUpdatedDate="Now" IObject_CreationUser="superuser"
      IObject_DomainUID="Auto" IObject_UniqueKey="Auto" IRel_DomainUID1="Auto"
      IRel_DomainUID2="Auto">
- <Rel>
  <IRel UID1="EngineeringDocVersion" UID2="FRM_DesignDocument-Form"
      DefUID="SPFClassDefForm" IsRequired="False" OrderValue="0" />
  <IObject UID="EngineeringDocVersion.DesignDocument-Form" />
  </Rel>
- <Rel>
  <IRel UID1="EDC_Engineering_Document_Types_1" UID2="EngineeringDocVersion"
      DefUID="EngineeringDocClassClassDef" IsRequired="False" OrderValue="0" />
  <IObject UID="EDC_Engineering_Document_Types_1.EngineeringDocVersion" />
  </Rel>
  </Container>
```

6.4 Creating Methods for Classified Objects

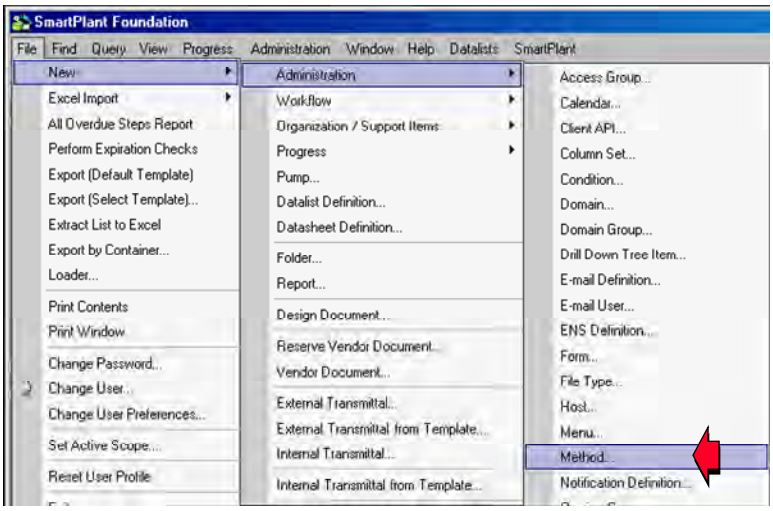
The next step is to continue to configure the GUI and place a new menu option under the ***File*** > ***New*** menu for an *Engineering Document*. This will require new methods and menu items.

To configure a new method:

INTERGRAPH

Creating Methods

- **From the menu, select *File> New > Administration > Method.***

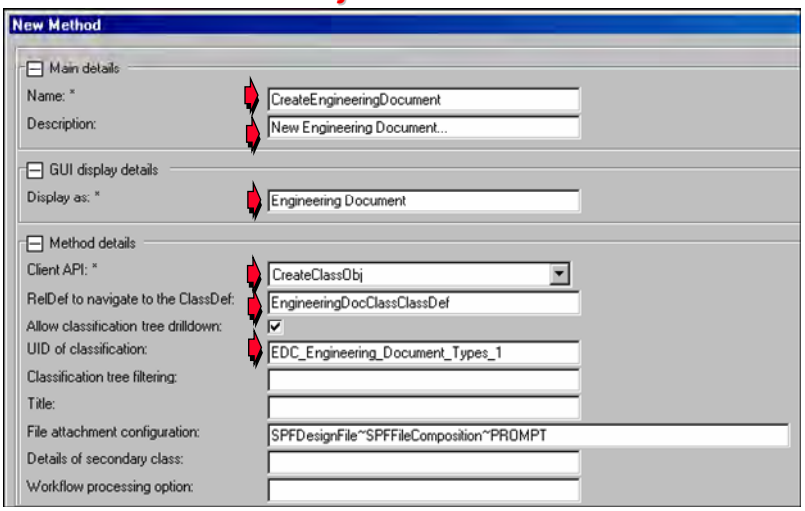


© 2008. Intergraph Corp.
All Rights Reserved.

The *New Method* form will appear.

Creating Methods

- **Enter the necessary values to define the new method. Select *CreateClassObj* for the *Client API*.**



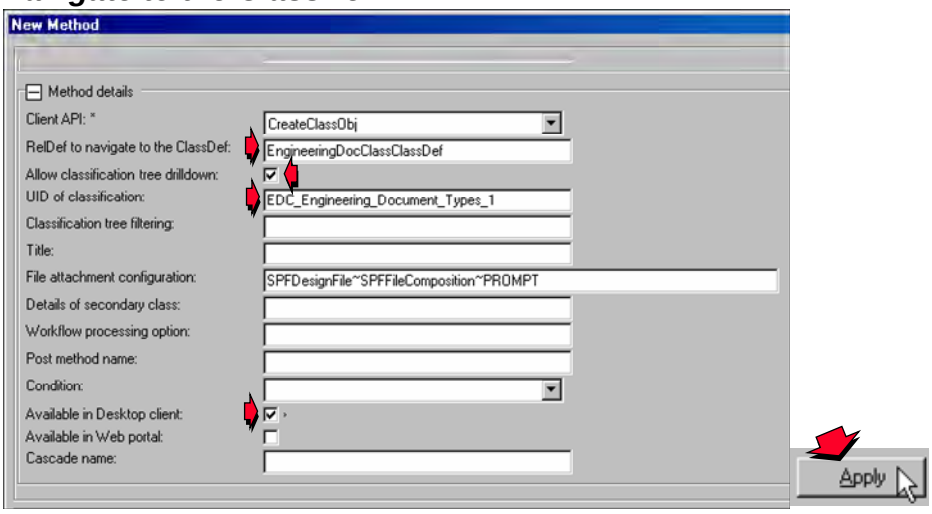
© 2008. Intergraph Corp.
All Rights Reserved.

The *New Method* form contains the following fields:

- **Name** - CreateEngineeringDocument
- **Description** - New Engineering Document...
- **Display as** - Engineering Document
- **Client API** - CreateClassObj
- **RelDef to navigate to the ClassDef** - EngineeringDocClassClassDef
- **Allow classification tree drilldown** - True (enabled)
- **UID of classification** - EDC_Engineering_Document_Types_1

Creating Methods

- **Enter *EngineeringDocClassClassDef* for the *RelDef to navigate to the ClassDef.***



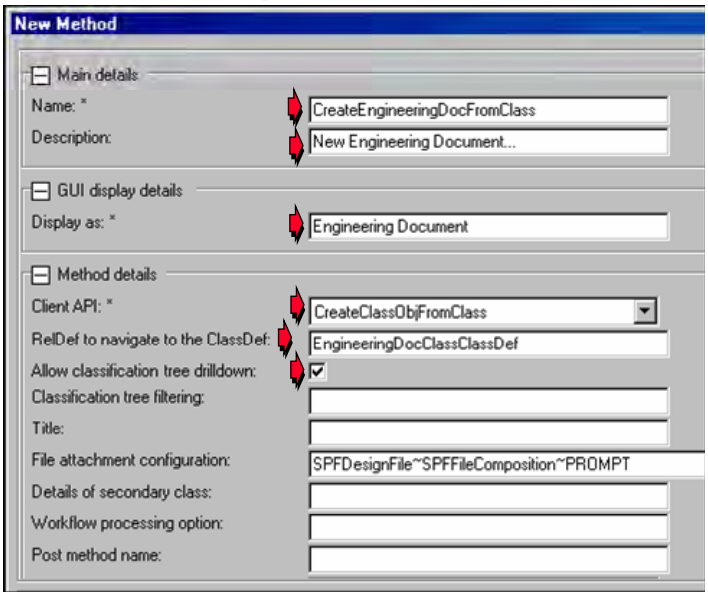
© 2008. Intergraph Corp.
All Rights Reserved.

The *New method* form fields (con't):

- **Available in Desktop client** - True (enabled)

Repeat the steps to define another method.

Creating Methods

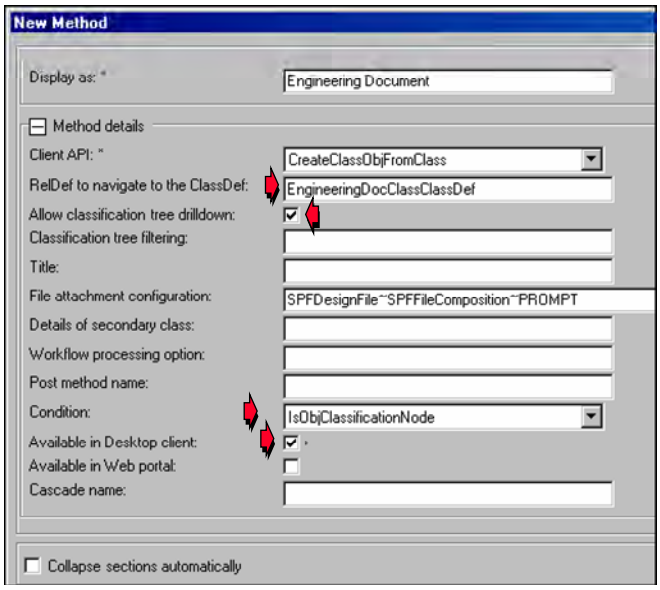


- **Enter the necessary values to define the new method. Select *CreateClassObjFromClass* for the *Client API*.**

The *New Method* form contains the following fields:

- **Name** - CreateEngineeringDocFromClass
- **Description** - New Engineering Document...
- **Display as** - Engineering Document
- **Client API** - CreateClassObjFromClass
- **RelDef to navigate to the ClassDef** - EngineeringDocClassClassDef
- **Allow classification tree drilldown** - True (enabled)
-

Creating Methods



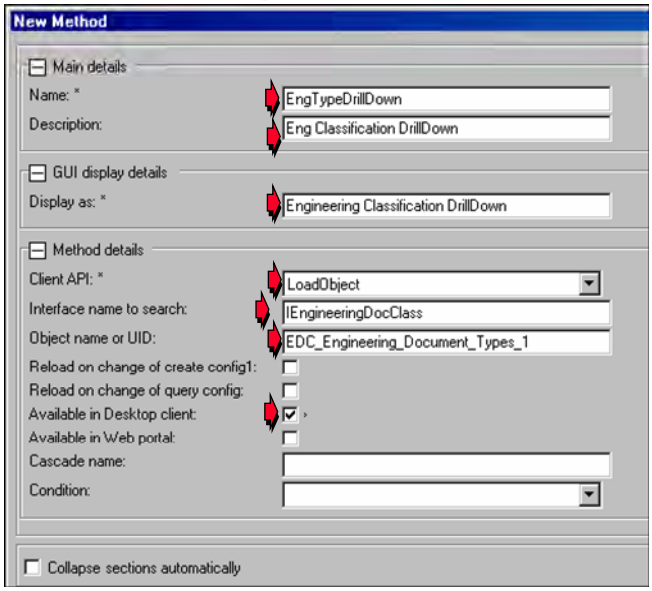
- **Enter *EngineeringDocClass ClassDef* for the *RelDef to navigate to the ClassDef.***

The *New method* form fields (con't):

- **Condition** - IsObjClassificationNode
- **Available in Desktop client** - True (enabled)

Repeat the steps to define a tree drill down method.

Creating Methods



- **Enter the necessary values to define the new method. Select *LoadObj* for the *Client API*.**

The *New Method* form contains the following fields:

- **Name** - EngTypeDrillDown
- **Description** - Eng Classification DrillDown
- **Display as** - Engineering Classification DrillDown
- **Client API** - LoadObject
- **Interface name to search** - IEngineeringDocClass
- **Object name or UID** - EDC_Engineering_Document_Types_1
- **Available in Desktop client** - True (enabled)

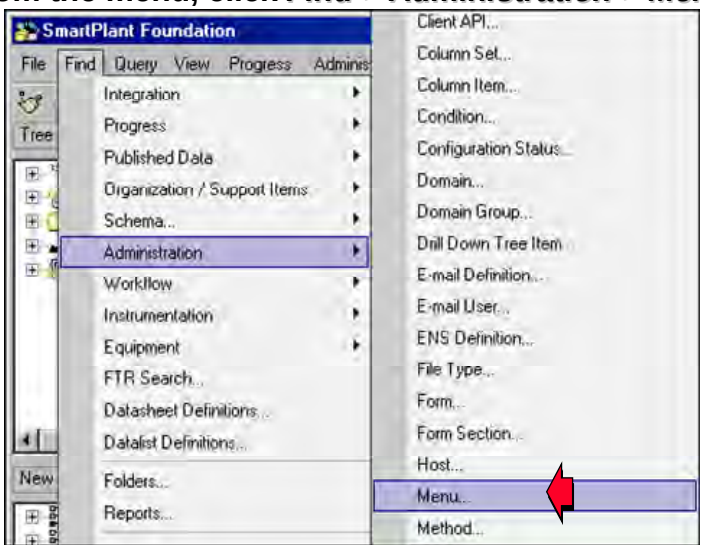
6.5 Creating Menu Items for Classified Objects

To use the new classification tree, new menu items must be defined for the ***File*** > ***New*** menu.

Perform a search for the *New* menu node in order to add the ability to create a new Engineering Document.

Managing Menu Items

- **From the menu, click *Find* > *Administration* > *Menu*.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Menu* dialog will appear.

Managing Menu Items

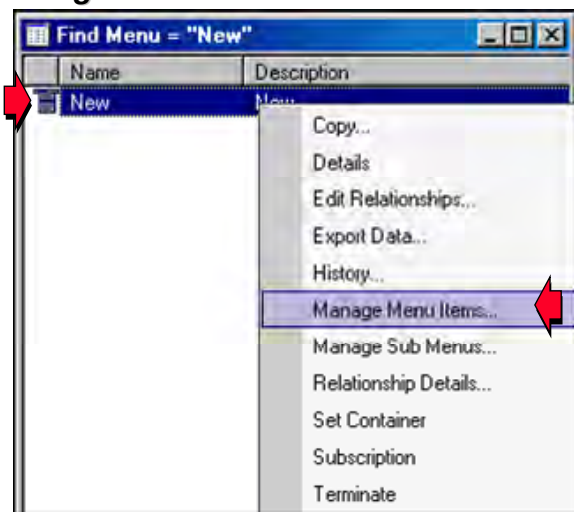
- **Enter the criteria find and display all Menus that contain *New* and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Locate the *New* menu node in the list view.

Managing Menu Items

- **Right click on the *New* menu and from the shortcut menu, select *Manage Menu Items*.**

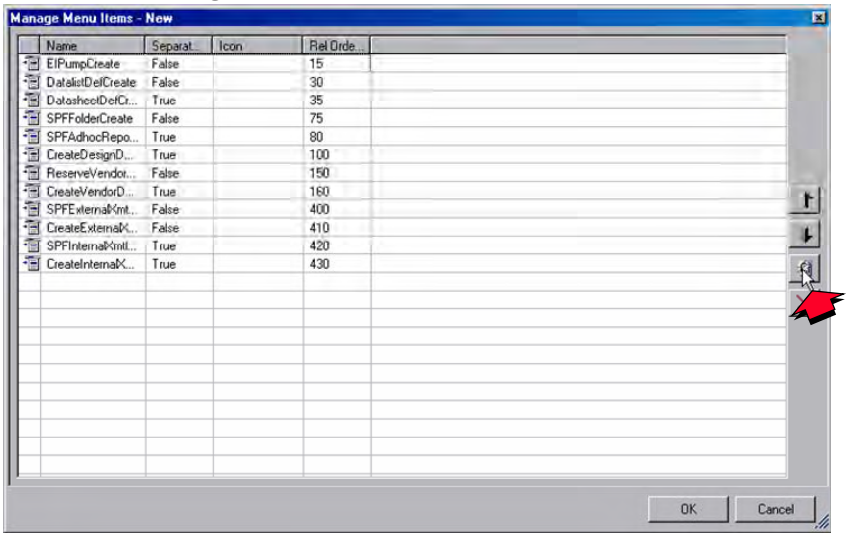


© 2008. Intergraph Corp.
All Rights Reserved.

Start from the *Manage Menu Items* form by clicking the **Create** button.

Creating Menu Items

- **On the *Manage Menu Items* form, click the *Create* button.**

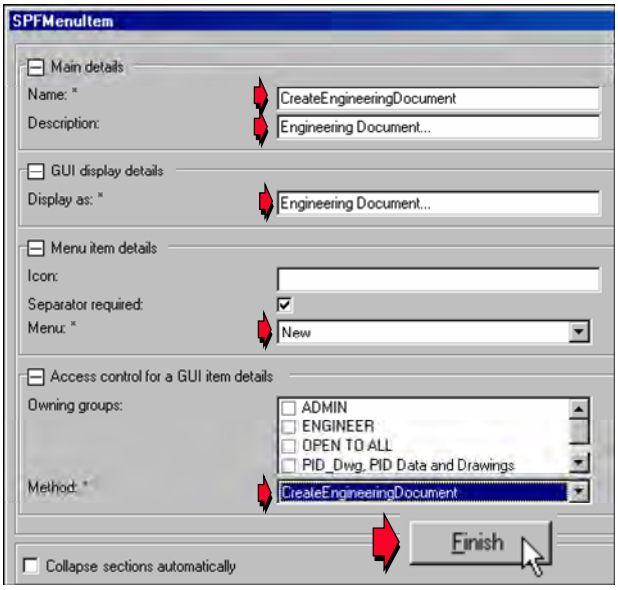


© 2008. Intergraph Corp.
All Rights Reserved.

Enter the required values to define a menu item to create a new Engineering Document.

Creating Menu Items

- **Enter appropriate values to define a new *Menu Item* and click Finish.**

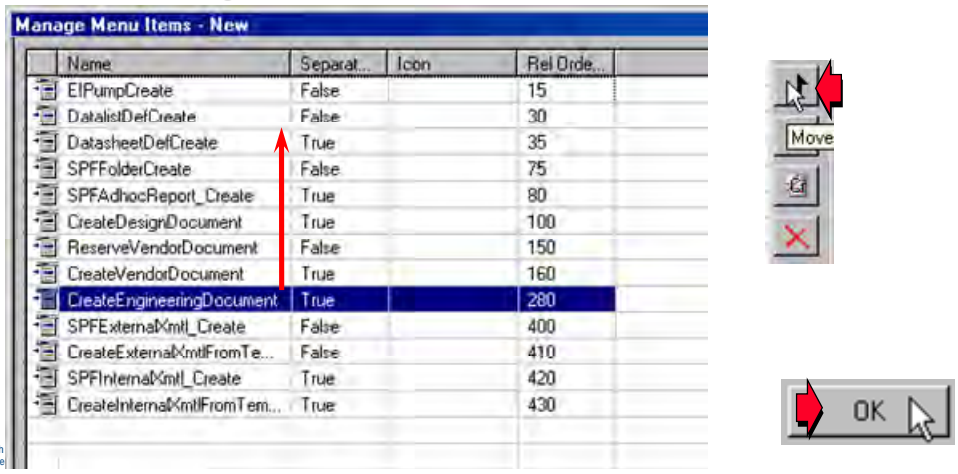


© 2008. Intergraph Corp.
All Rights Reserved.

To change the sequence of an item in the menu, select the item and use the *up* and *down* arrow buttons on the form to move the item.

Managing Menu Items

- **Select the *Menu Item* and click the up arrow button to move *CreateEngineeringDocument* towards the top of the menu then click OK.**



© 2008. Intergraph
All Rights Reserve

The Engineering Document item has been added to the ***New*** menu.

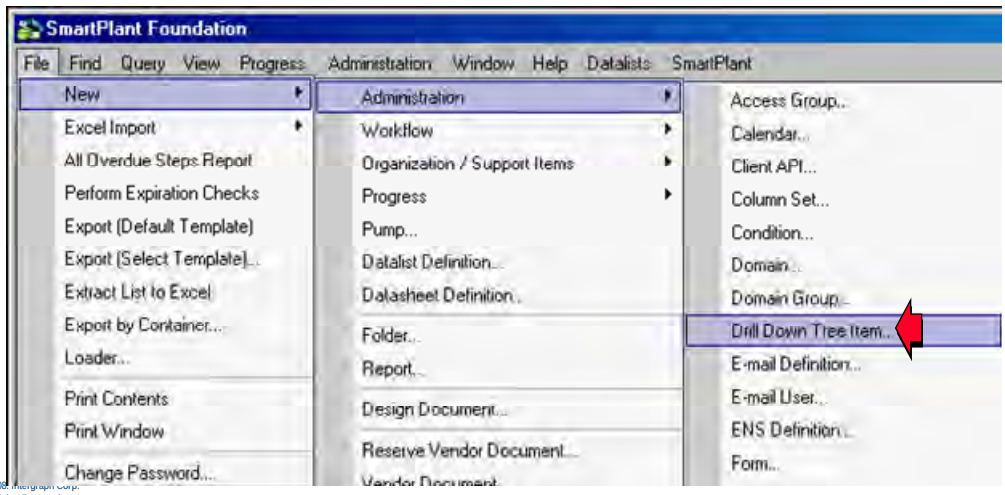
6.6 Creating a Drill Down Tree

By default, the Desktop Client will display the *Document category* list in the Tree View. This means that any document objects created will be associated to this tree structure if you drill down (expand) the Document category tree.

In this section, you will see how to create and configure a new drill down tree for the new Engineering Document Types hierarchy.

Creating Drill Down Tree

- **From the menu, select *File > New > Administration > Drill Down Tree Item.***

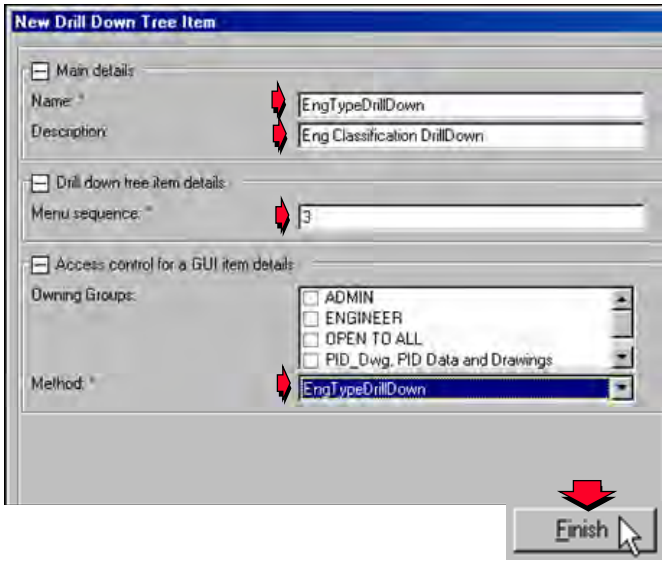


© 2008 Intergraph Corp.
All Rights Reserved.

The *New Drill Down Tree Item* form is displayed.

Creating Drill Down Tree

- **Enter the values to create a new *Drill Down Tree Item* and click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *New Drill Down Tree Item* form contains the following fields:

- **Name** - specifies the name/purpose of the drill down tree and is a required field.
- **Description** - specifies a description for the drill down tree and is an optional field.
- **Menu sequence** - defines a sequence number for the drill down tree item.
- **Owning groups** - select the owning group that can access this drill down tree item. Selecting an owning group will restrict who can access the item. This is not to be confused with access group control that is configured on the related method. This is an optional field.
- **Method** - select a method from the list. The method must have an access group and is a required field.

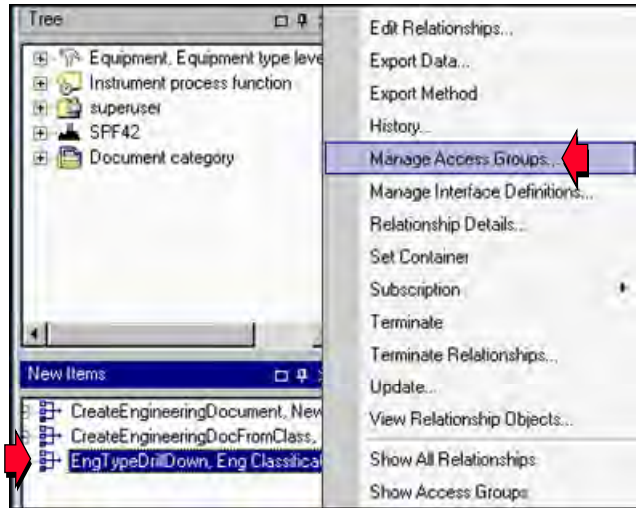
6.6.1 Adding Method Access Rights

Right click on *EngTypeDrillDown* and from the shortcut menu, select ***Manage Access Groups***.

INTERGRAPH

Manage Related Items

- **Right click on the *EngTypeDrillDown* and from the shortcut menu, select *Manage Access Groups*.**

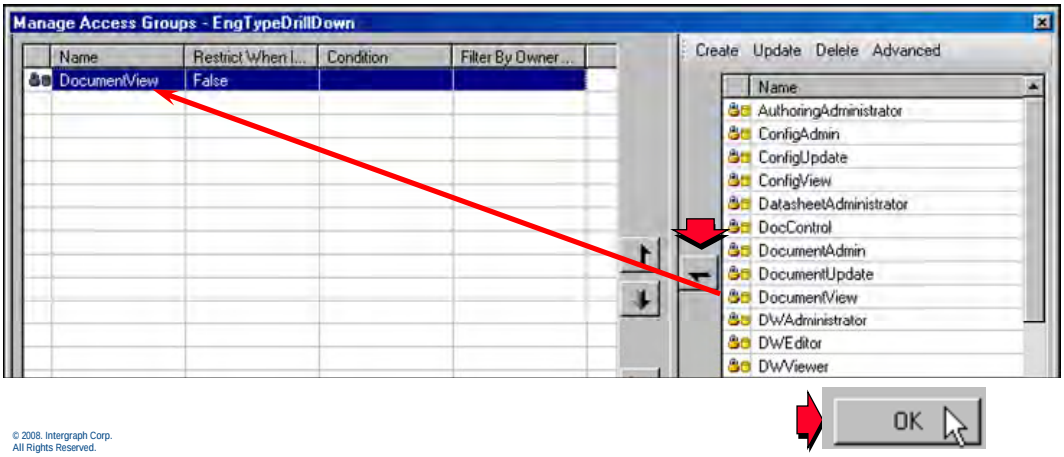


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Access Groups* form appears.

Manage Related Items

- **Select *DocumentView* and click the left arrow button to relate this access group to *EngTypeDrillDown then* click OK.**

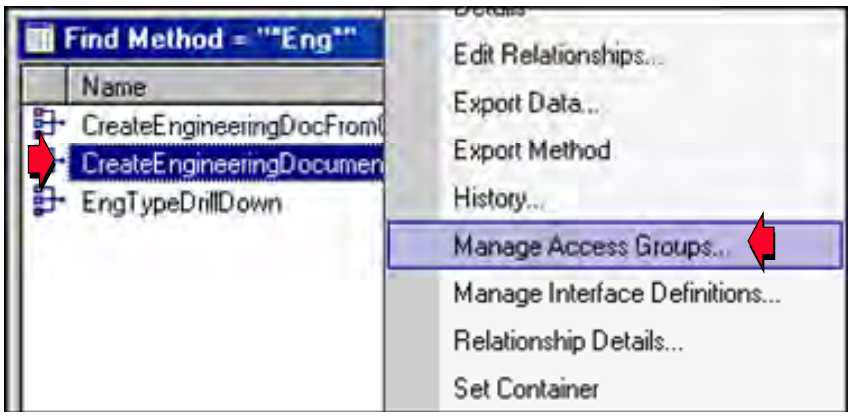


© 2008. Intergraph Corp.
All Rights Reserved.

Repeat the steps to do the same thing with the *CreateEngineeringDocument* method. Right click on *CreateEngineeringDocument* and from the shortcut menu, select ***Manage Access Groups***.

Manage Related Items

- **Right click on the *EngTypeDrillDown* and from the shortcut menu, select *Manage Access Groups*.**

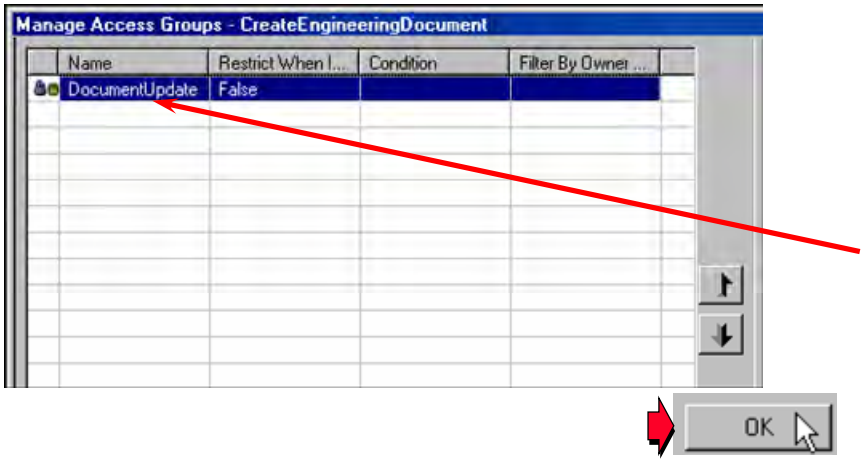


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Access Groups* form appears.

Manage Related Items

- **Select *DocumentUpdate* and click the left arrow button to relate this access group to *CreateEngineeringDocument then* click OK.**

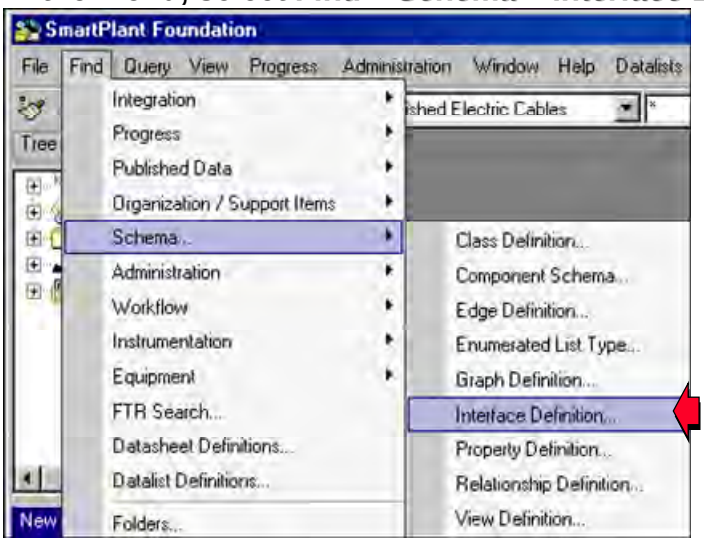


© 2008. Intergraph Corp.
All Rights Reserved.

Add method access to the *IEngineeringDoc* interface.

Manage Related Items

- **From the menu, select *Find > Schema > Interface Definition.***

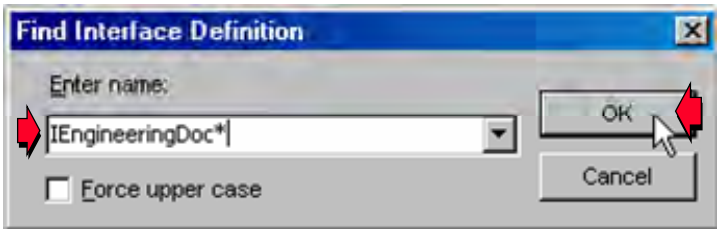


© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Interface Definition* dialog will appear.

Manage Related Items

- **Enter *IEngineeringDoc**(with the wildcard character) to find interface definitions that start with *IEngineeringDoc* and click OK.**

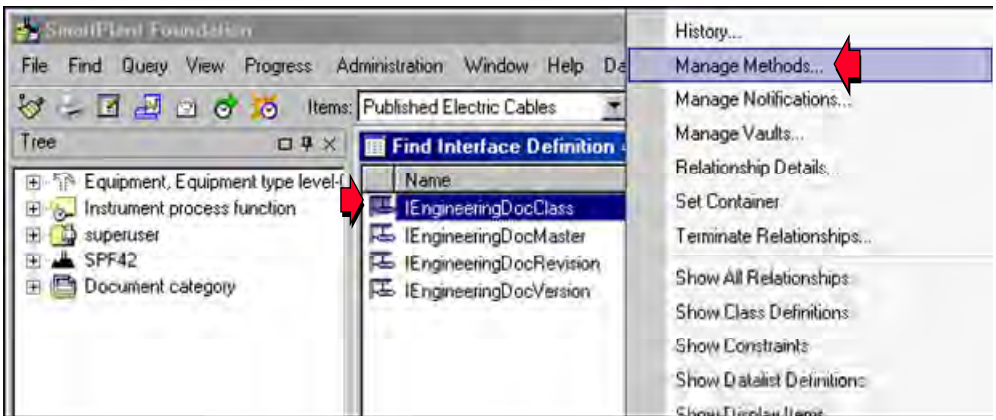


© 2008. Intergraph Corp.
All Rights Reserved.

Use the *Manage Methods* form to relate the new methods to an interface.

Manage Related Items

- **Right click on *IEngineeringDocClass* and from the shortcut menu, select *Manage Methods*.**

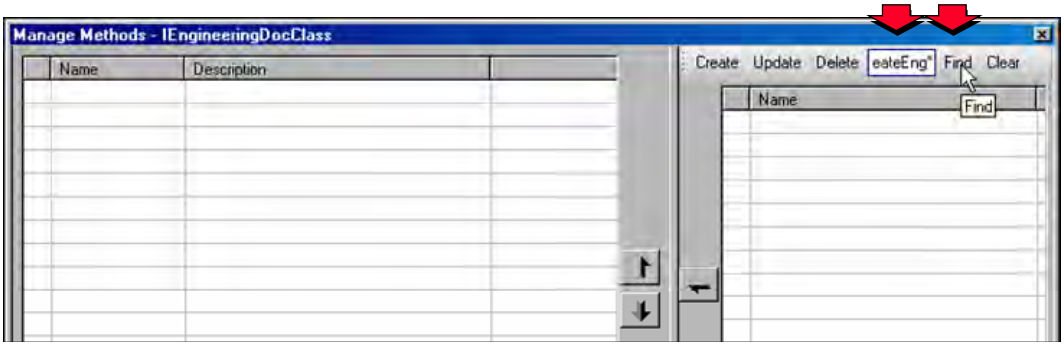


© 2008. Intergraph Corp.
All Rights Reserved.

The *Manage Methods* form appears.

Manage Related Items

- **To find the method, enter *CreateEng** in the search field and click Find.**

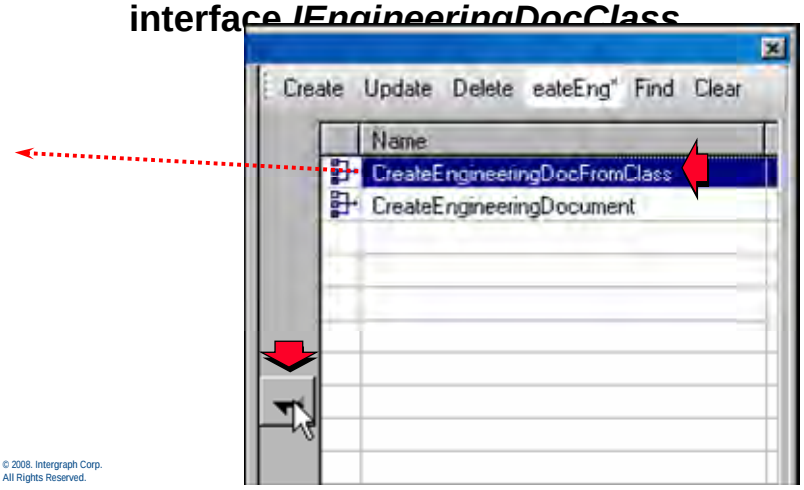


© 2008. Intergraph Corp.
All Rights Reserved.

The ***CreateEngineeringDocFromClass*** method (created earlier) will appear on the right side of the form.

Manage Related Items

- **Select *CreateEngineeringDocFromClass* and click the left arrow button to relate the item to the interface *IEngineeringDocClass***

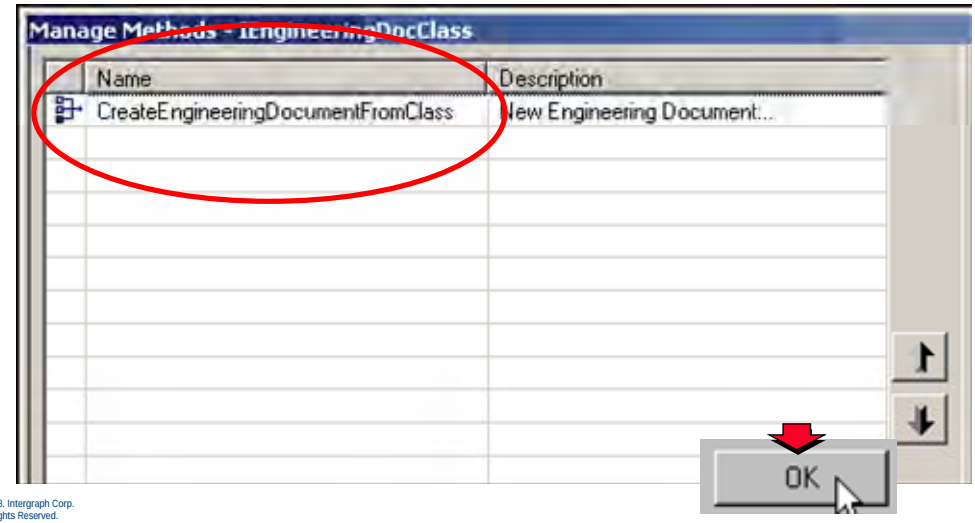


© 2008. Intergraph Corp.
All Rights Reserved.

This update is saved to the database on selection of **OK**.

Manage Related Items

- **To save the relationship, click OK.**



© 2008. Intergraph Corp.
All Rights Reserved.

The menu commands to create classified Engineering Document objects have now been configured.

6.7 Configuring the Engineering Numbering System

The Engineering Numbering System (ENS) provides an automated object identification facility, which allows project-specific identification formats to be configured as "templates". SPF then uses these templates to guide the user when creating new objects. The **ENS** definition is composed of various components that can extract data from other display items on the form as well as constants. These components are configured in sequence with additional delimiter characters to define the required format.

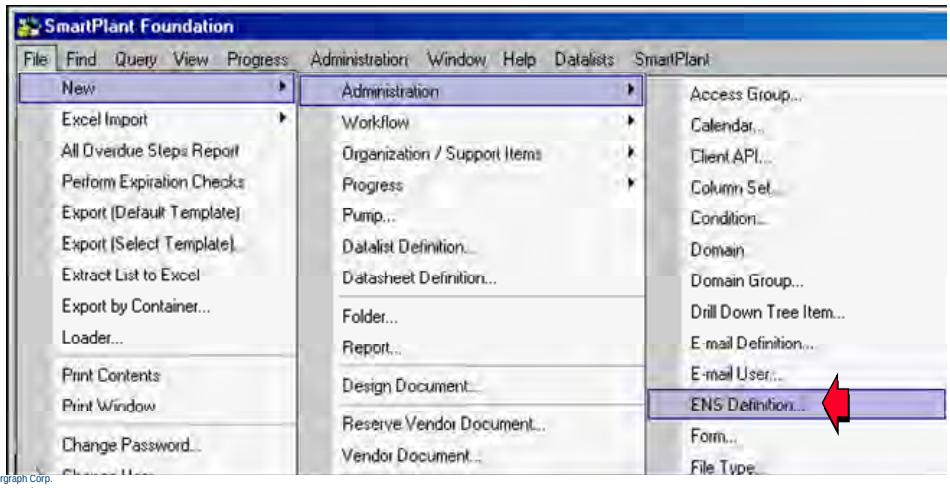
In Engineering Document Type example, the Engineering Document object name will be determined by using the ENS. It will consist of a combination of a constant prefix and a 5-digit sequence.

Begin by defining an ENS definition object.

INTERGRAPH

Engineering Numbering System

- **Select *File > New > Administration > ENS Definition* from the menu.**

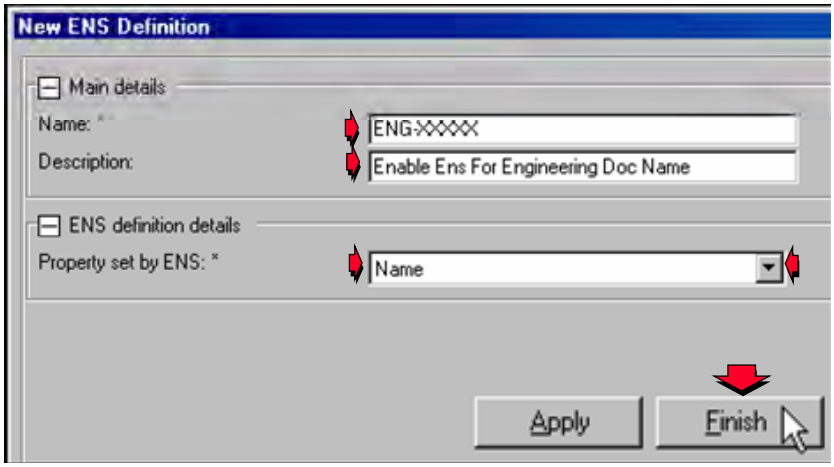


© 2008. Intergraph Corp.
All Rights Reserved.

The *New ENS Definition* form will appear.

Engineering Numbering System

- **Enter a required name, optional description and select a property def to associate with this ENS Def, then click Finish.**



© 2008. Intergraph Corp.
All Rights Reserved.

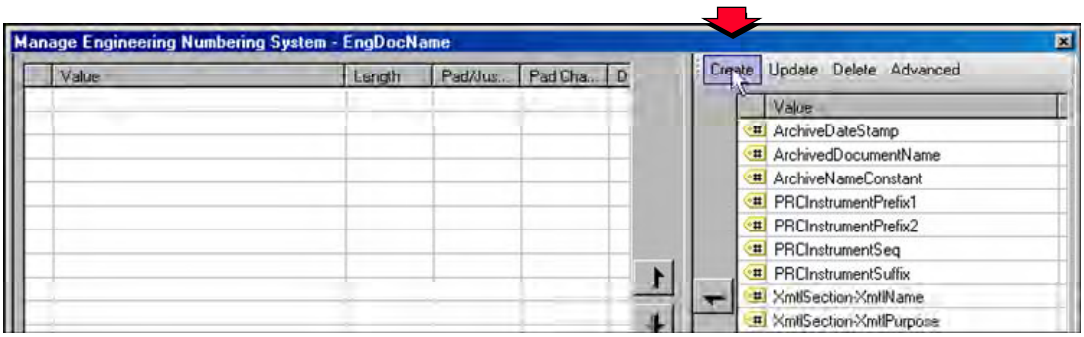
The *New ENS Definition* form contains the following fields:

- **Name** - specifies the name/purpose of the ENS definition and is a required field.
- **Description** - specifies a description for the ENS definition and is an optional field.
- **Property set by ENS** - specifies. This is a required field.

The *Manage Engineering Numbering System* form will automatically appear.

Engineering Numbering System

- **Select the Create command to create a new ENS Item.**

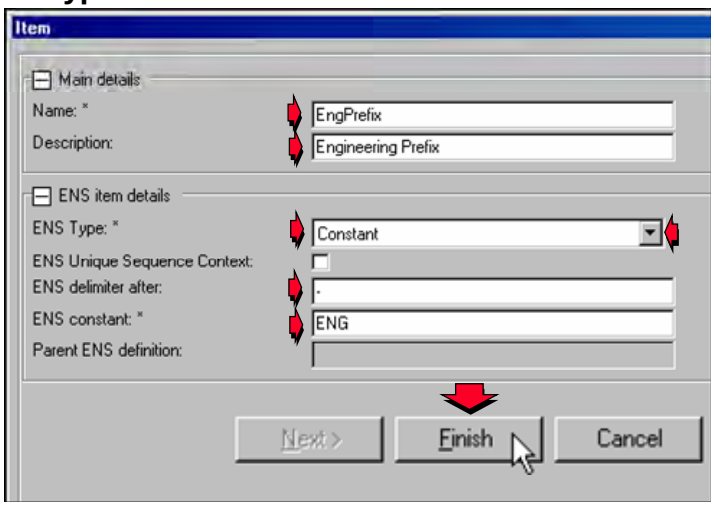


© 2008. Intergraph Corp.
All Rights Reserved.

The *New ENS Item* form will appear.

Engineering Numbering System

- **Enter a required name, optional description and select a ENS Type to associate with this ENS Def.**



© 2008. Intergraph Corp.
All Rights Reserved.

The *New ENS Item* form contains the following fields:

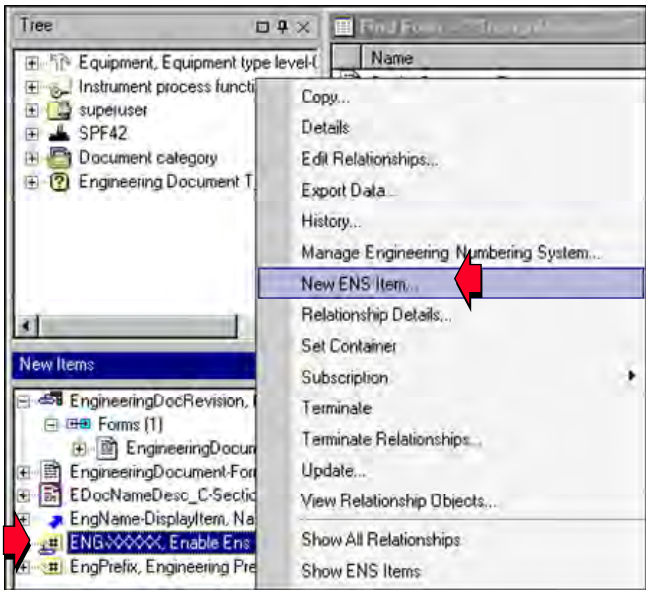
- **Name** - specifies the name/purpose of the ENS item and is a required field.
- **Description** - specifies a description for the ENS item and is an optional field.
- **ENS Type** - specifies that an ENS Item can be one of the following types and is a required field:
  - **Function** - used for automatic Sequence numbers
  - **Property** - specifies that this is going to be a property def
  - **Related Item** - specifies that this is going to be a RelDef or EdgeDef
  - **Constant** - specifies a text value
- **ENS Unique Sequence Context** - specifies that the sequence number used will be defined in the context of the other items in the ENS definition.
- **ENS delimiter after** - specifies the delimiter to add when it builds the ENS string.
- **Parent ENS definition** - specifies the ENSDef the item is related to.

ENS Items can also be created from the right mouse menu.

INTERGRAPH

Engineering Numbering System

- **Right click on *ENG-XXXXX* and from the shortcut menu, select *New ENS Item*.**

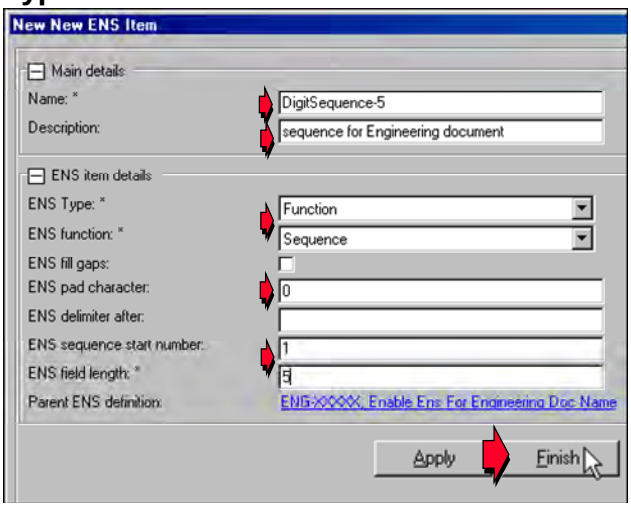


© 2008. Intergraph Corp.
All Rights Reserved.

The *New ENS Item* form will appear.

Engineering Numbering System

- **Enter a required name, optional description and select a ENS Type to associate with this ENS Def.**



© 2008. Intergraph Corp.
All Rights Reserved.

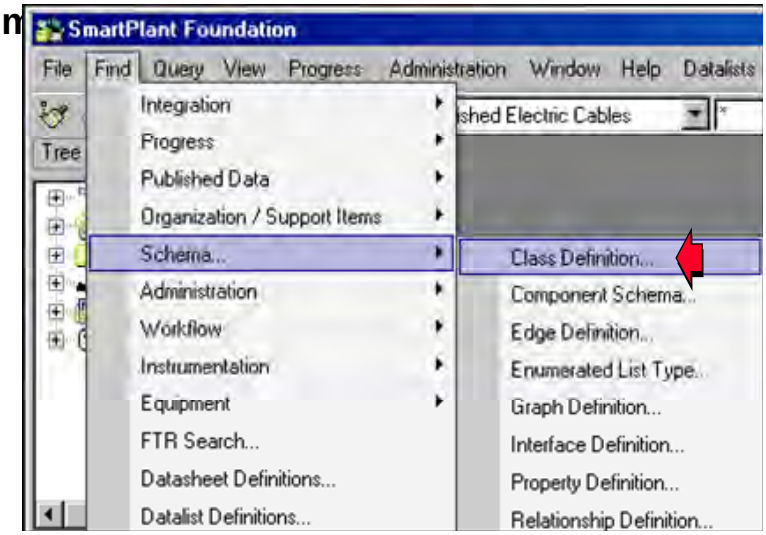
6.7.1 Relating the ENS to the Classification

Once the ENS Definition has been created and configured, it must be related to the EngineeringDoc class.

INTERGRAPH

Engineering Numbering System

- **Select *Find > Schema > Class Definition* from the m**



© 2008. Intergraph Corp.
All Rights Reserved.

The *Find Class Definition* dialog will appear.

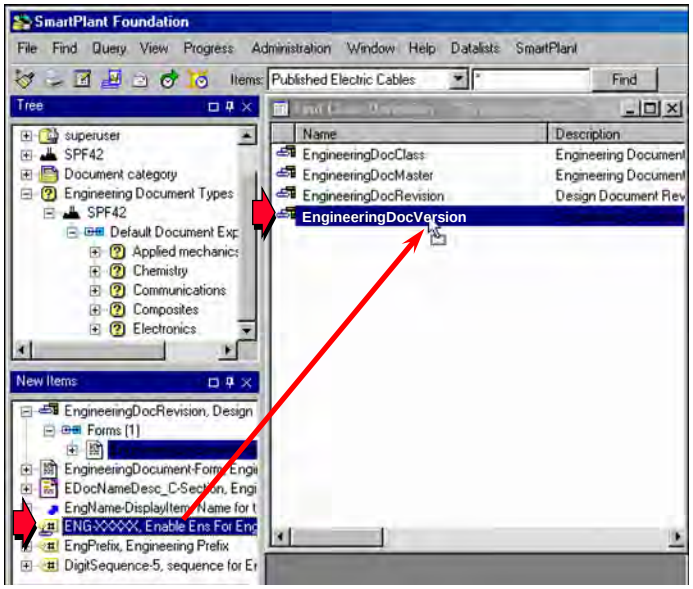
Engineering Numbering System

- **Enter the criteria find and display the ClassDef that contains *EngineeringDoc* and click OK.**

© 2008. Intergraph Corp.
All Rights Reserved.

Relate the new ENS Definition to a class definition using drag and drop.

Engineering Numbering System

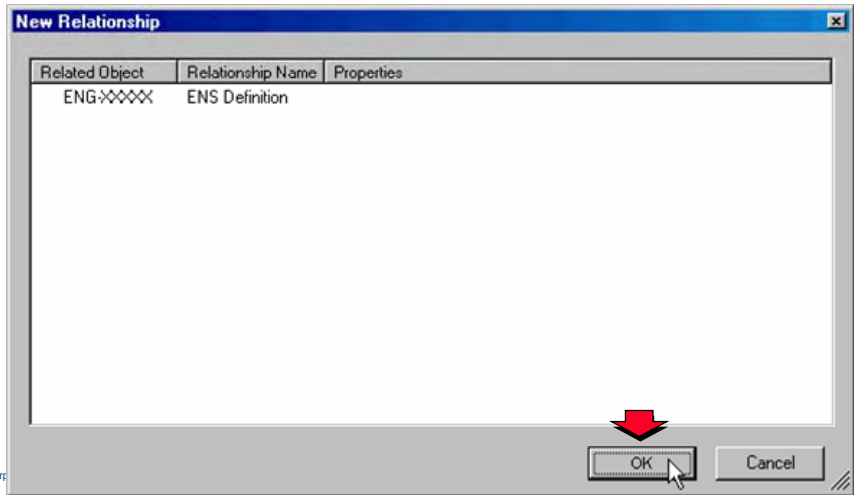


- **Drag and drop the *ENG-XXXXX* ENS Def onto the *EngineeringDoc Version* class.**

The *New Relationship* form will appear.

Engineering Numbering System

- **Click OK to create the new ENS (ENS Def → ClassDef) relationship.**



© 2008. Intergraph Corp
All Rights Reserved.

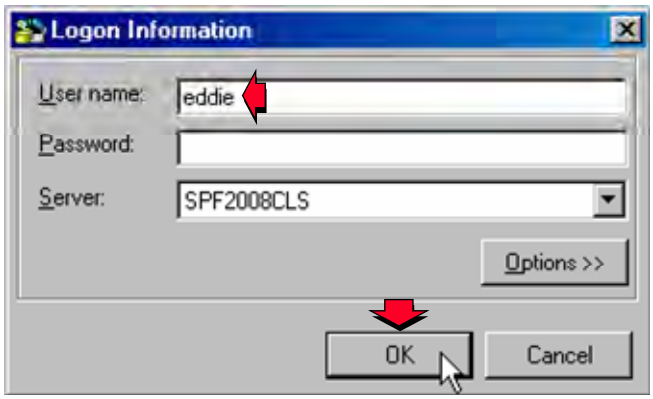
6.8 Testing Homogeneous Classifications

After the schema for the classifications are loaded and the GUI has been configured, log in as a create user to test the new drill down tree.

Start up the Desktop Client:

Testing Homogeneous Classifications

- **Enter a user name to log in and click OK.**

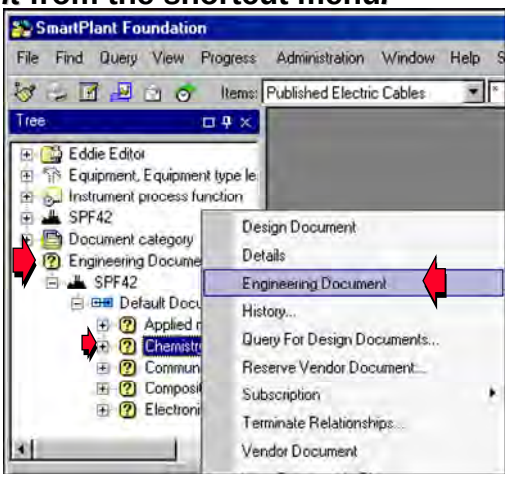


© 2008. Intergraph Corp.
All Rights Reserved.

Expand the Tree to see the new classification drill down tree.

Testing Homogeneous Classifications

- **Drill down in the Engineering Document Types tree, right click on the *Chemistry option* and select *Engineering Document* from the shortcut menu.**

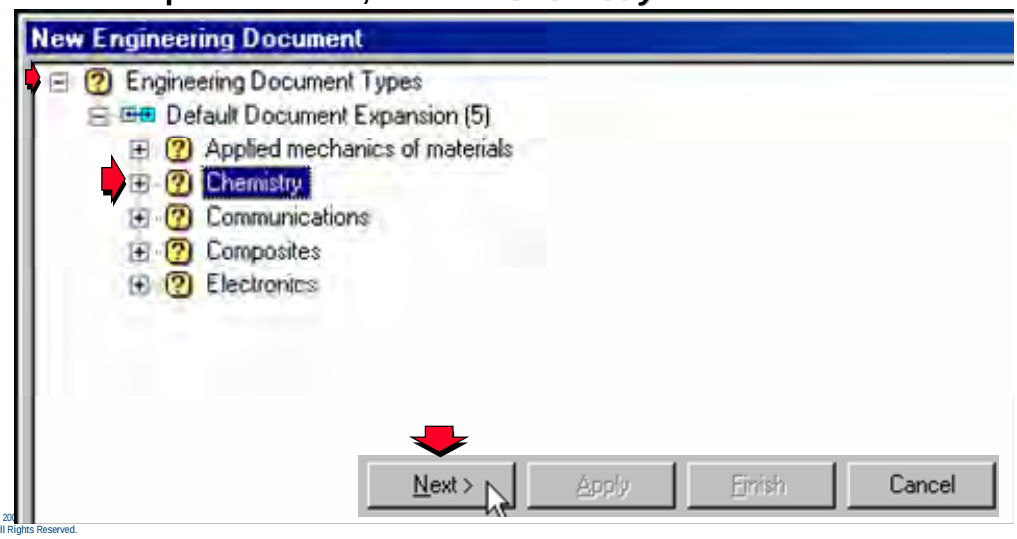


© 2008. Intergraph Corp.
All Rights Reserved.

In the tree view, click the classification for the new Engineering Document. For example, to create a Chemistry document, click *Engineering Document Types* > *Chemistry* in the tree view.

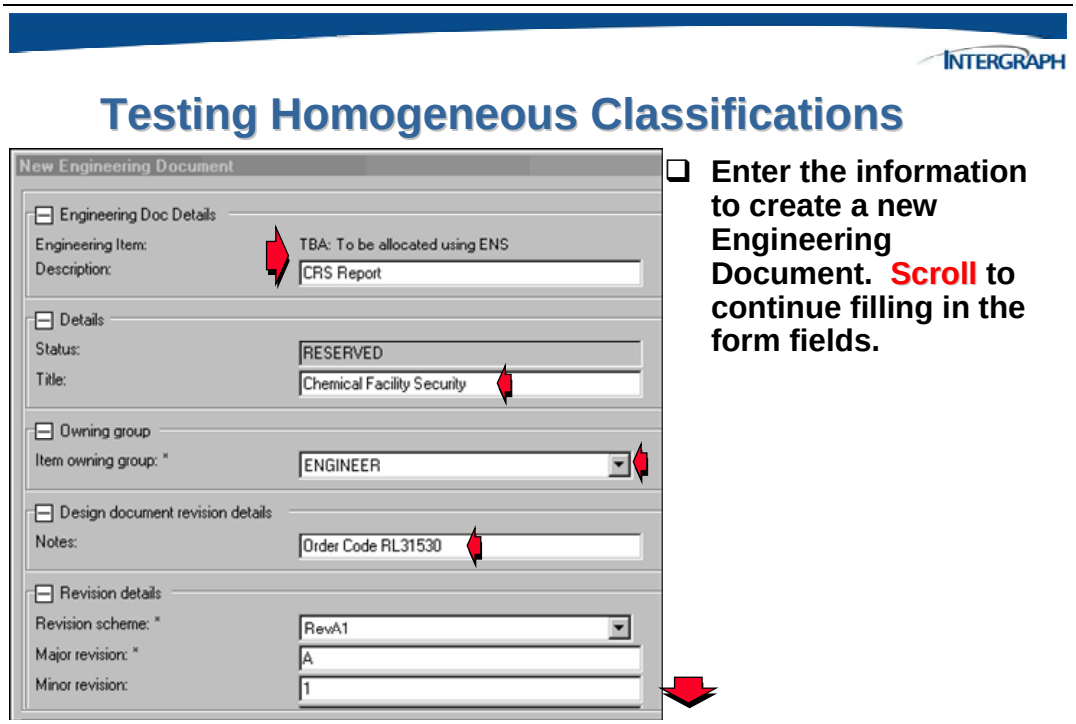
Testing Homogeneous Classifications

- **Expand the tree, click on *Chemistry* and select next.**

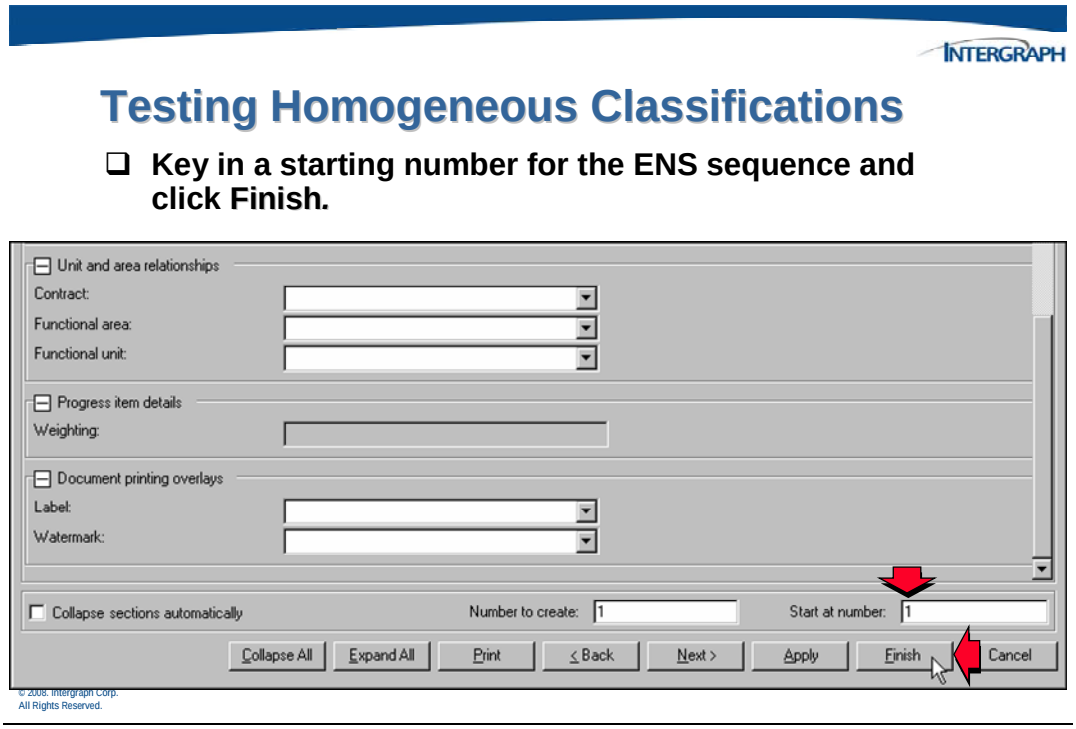


© 20
All Rights Reserved.

Define details for the new object, such as the description, title, item owning group, and notes for the document. Note the *name* field is labeled **Engineering Item** and will be allocated by the configured ENS.



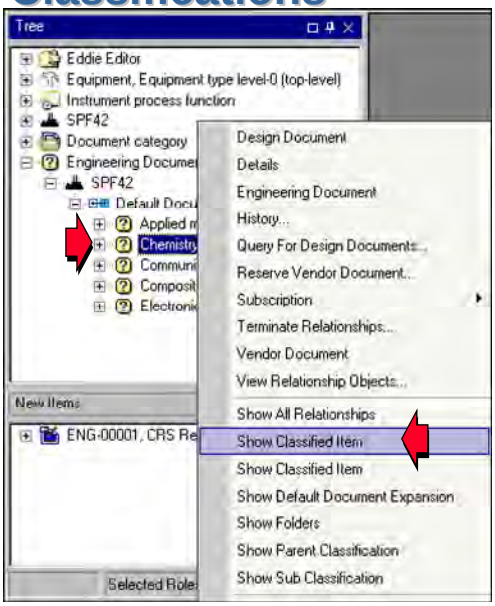
Scroll down to finish entering the necessary property values for this object instance.



The new engineering document, ENG-00001, will appear in the *New Items* window.

Testing Homogeneous Classifications

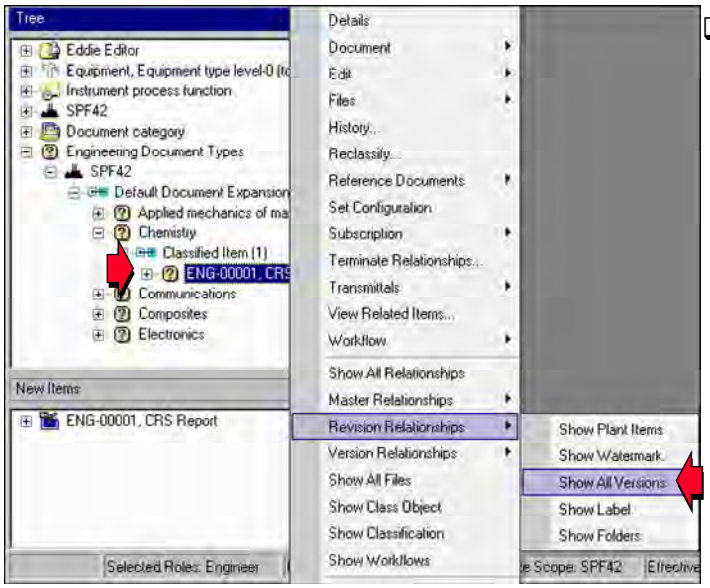
- **Right click on the *Chemistry* Engineering Document Type and select *Show Classified Item* from the shortcut menu.**



© 2008. Intergraph Corp.
All Rights Reserved.

Continue to drill down in the classified objects tree to view the related document version.

Testing Homogeneous Classifications

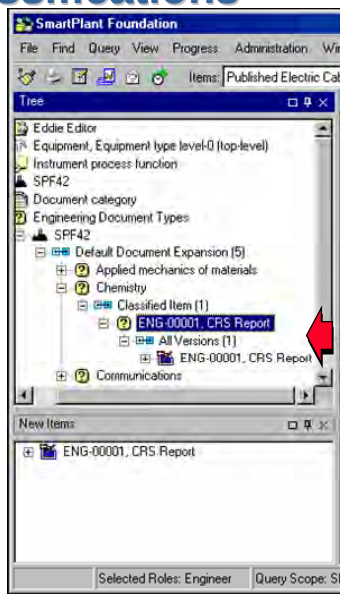


- **Right click on the new classified item and select *Revision Relationships > Show All Versions* from the shortcut menu.**

The results are displayed in the expanded classified objects tree.

Testing Homogeneous Classifications

- **The *Engineering Document Types* tree will be expanded to show the new related Chemistry document.**



© 2008. Intergraph Corp.
All Rights Reserved.

6.9 Configuring and Testing Heterogeneous Classifications

The current configuration is homogeneous meaning all of the engineering documents will be created off the same ClassDef. to make a heterogeneous configuration document objects will be created related to different ClassDefs. The relationship between the classification selected and the ClassDef determines the object to be created.

Now the configuration is such that the relationship defined between the classification at the top of the tree and the *EngineeringDocVersion*.

```
<Rel>
<IRel UID1="EDC_Engineering_Document_Types_1" UID2="EngineeringDocVersion"
   DefUID="EngineeringDocClassClassDef" IsRequired="False" OrderValue="0" />
<IObject UID="EDC_Engineering_Document_Types_1.EngineeringDocVersion"  />
</Rel>
```

Which means for any classification selected, the software will walk up the classification tree looking for a RelDef like this.

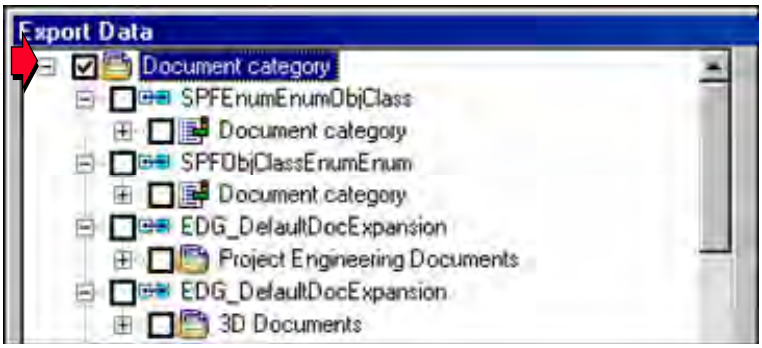
To make this heterogeneous another ClassDef (for master, revision and version) would need to be defined as well as a RelDef (like the example above) at a lower level in the tree.

As a guide, right click on the Document category classification in the tree and use the ***Export Data*** command from the right mouse menu.

INTERGRAPH

Heterogeneous Classifications

- ❑ **Enable the toggle next to Document category.**

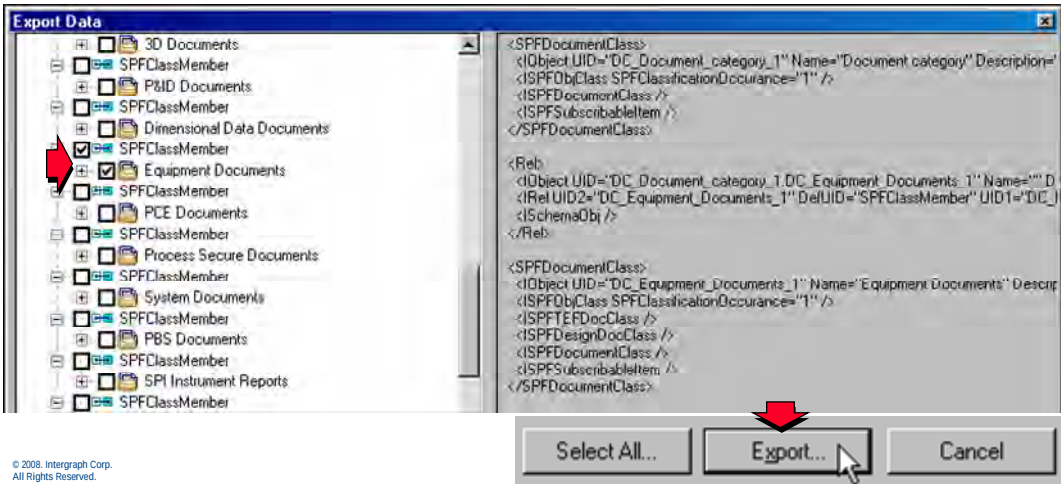


© 2008. Intergraph Corp.
All Rights Reserved.

As items are selected, you can see the xml statements in the right pane.

Heterogeneous Classifications

- **Scroll down and enable the toggles next to *SPFClassMember* and *Equipment Documents*.**



© 2008. Intergraph Corp.
All Rights Reserved.

Open the export file that is generated by the loader in order to create a new load file.

Heterogeneous Classifications

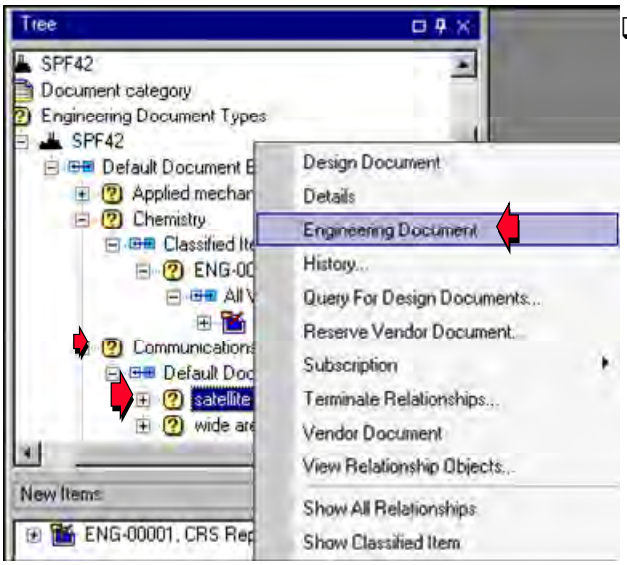
- **Copy, paste and edit the exported xml to create a new load file.**

```
- <Container CompSchema="MetaSchema" Scope="Schema"
    SoftwareVersion="04.02.00.04"
    IObject_ContainerID="Core.SPFDesignDocumentSchema"
    IObject_OBID="Auto" IObject_CreationDate="Now"
    IObject_TerminationDate="9999/12/31-23:59:59:999"
    IObject_LastUpdatedDate="Now" IObject_CreationUser="superuser"
    IObject_DomainUID="Auto" IObject_UniqueKey="Auto"
    IRel_DomainUID1="Auto" IRel_DomainUID2="Auto">
  - <Rel>
      <IRel UID1="EDC_satellite_1" UID2="SPFDesignDocVersion"
        DefUID="EngineeringDocClassClassDef" IsRequired="False"
        OrderValue="0" />
      <IObject UID="EDC_satellite_1.SPFDesignDocVersion" />
    </Rel>
  </Container>
```

© 2008. Intergr
All Rights Resi

In this example, the EDC_satellite_1 UID (satellite sub classification in the tree) is will be related to the DesignDocVersion class. Re-run the loader to load this new RelDef.

Testing Heterogeneous Classifications

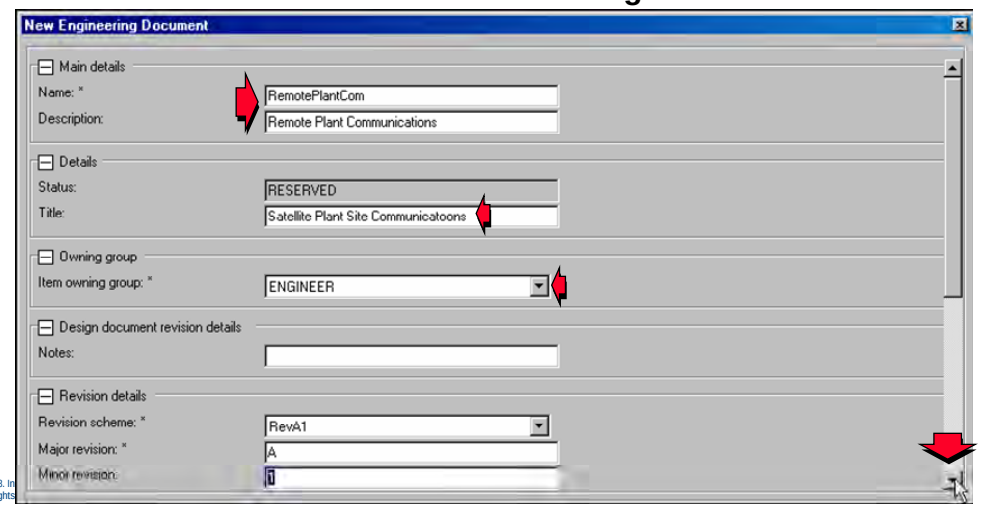


- **Right click on the *satellite* option in the Engineering Document Types tree and select *Engineering Document* from the shortcut menu.**

Define details for the new object, such as the name, description, title, item owning group, and notes for the document. Because this classification is related to a different class and a different form, the name field is not configured by the ENS.

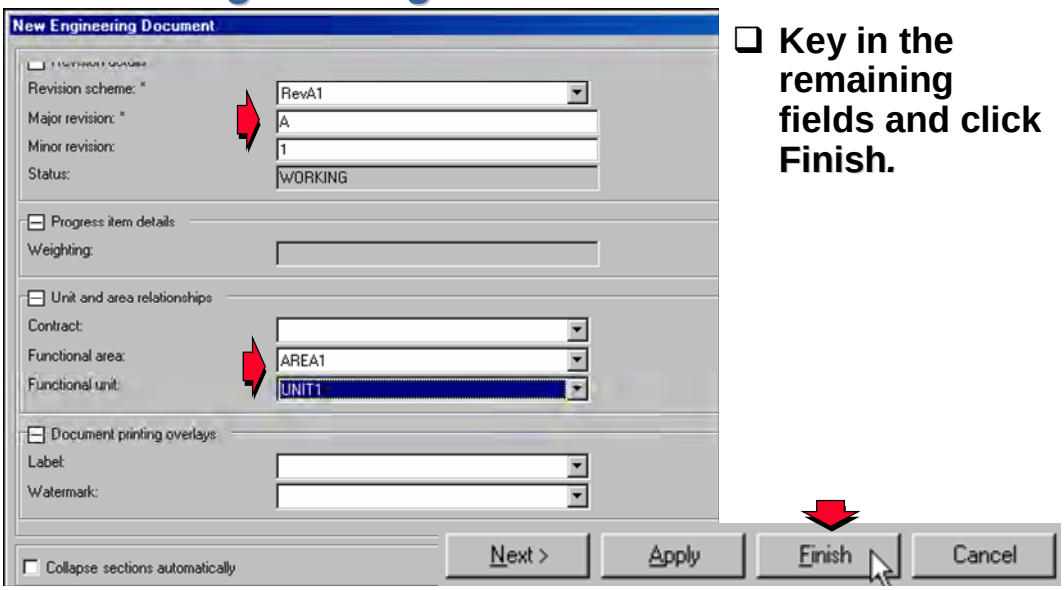
Testing Heterogeneous Classifications

- **Enter the information to create a new Engineering Document. Scroll to continue filling in the form fields.**



Since this is a lower level in the classification tree, it will have different behavior.

Testing Heterogeneous Classifications

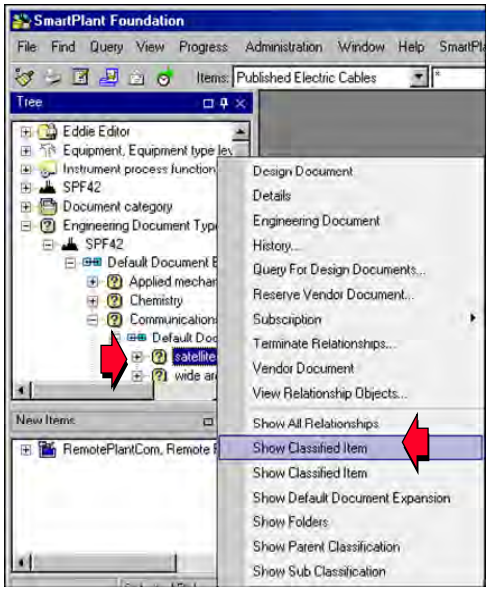


- **Key in the remaining fields and click Finish.**

The new engineering document, RemotePlantCom, will appear in the *New Items* window.

Testing Heterogeneous Classifications

- **Right click on the *satellite* Engineering Document Type and select *Show Classified Item* from the shortcut menu.**

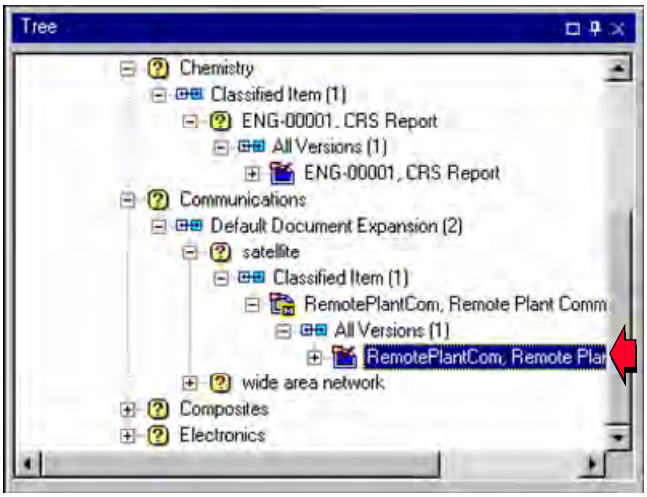


© 2008. Intergraph Corp.
All Rights Reserved.

Drill down in the classified objects tree to view the related document version..

Testing Heterogeneous Classifications

- **The *Engineering Document Types* tree will be expanded to show the new related Chemistry document.**



© 2008. Intergraph Corp.
All Rights Reserved.

The results are displayed in the expanded classified objects tree.

6.10 Activity – Creating Classified Objects

Complete the **Chapter 6 - Activity** in the SPF Configuration and Administration II activity workbook.

CHAPTER 7

SmartPlant Foundation Workflow

7. Introduction to SmartPlant Foundation Workflow

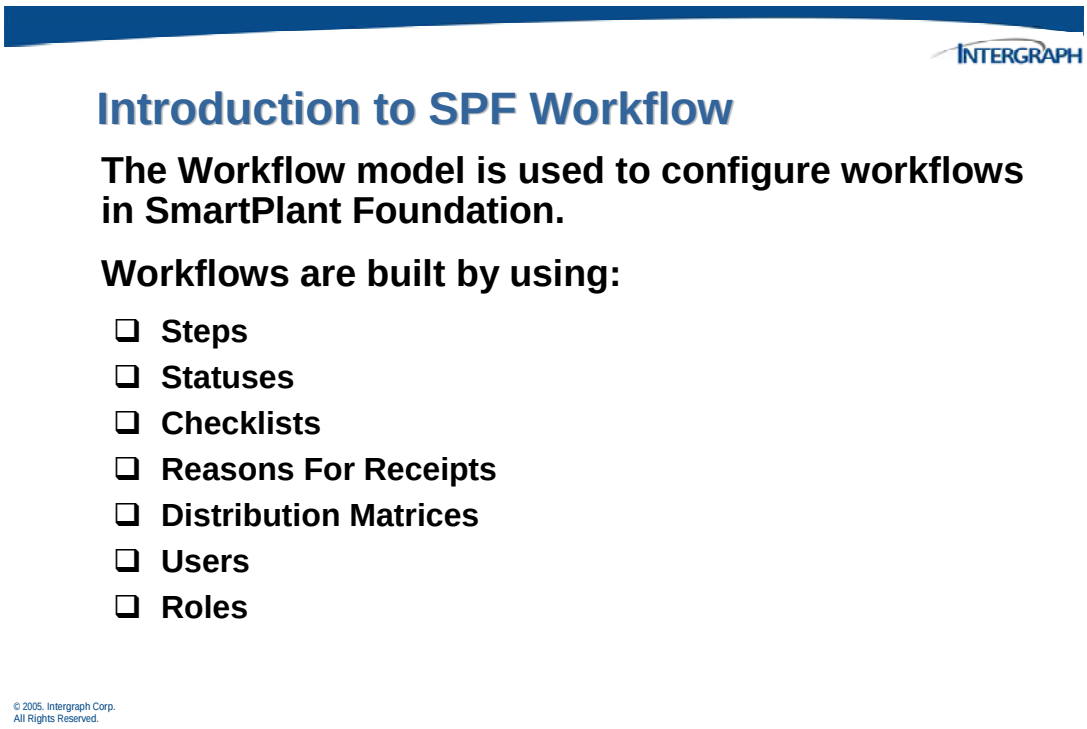
A workflow is a set of configured steps that are used to model and manage a customer's business processes such as the review and signoff of documents. The steps are configured to manage each stage of the process including *validation*, *approval*, *distribution*, and *notification*.

These steps can be performed in sequence or in parallel. The workflows branch depends on the outcome of the voting to specific steps in the same workflow or other workflows such as a generic rejection handling workflow.

Each step in a workflow has a target completion date, which is used to perform overdue notification and workflow expiration.

Any object can be passed down a workflow, but the most typical ones are:

- Document
- Folder
- Transmittal



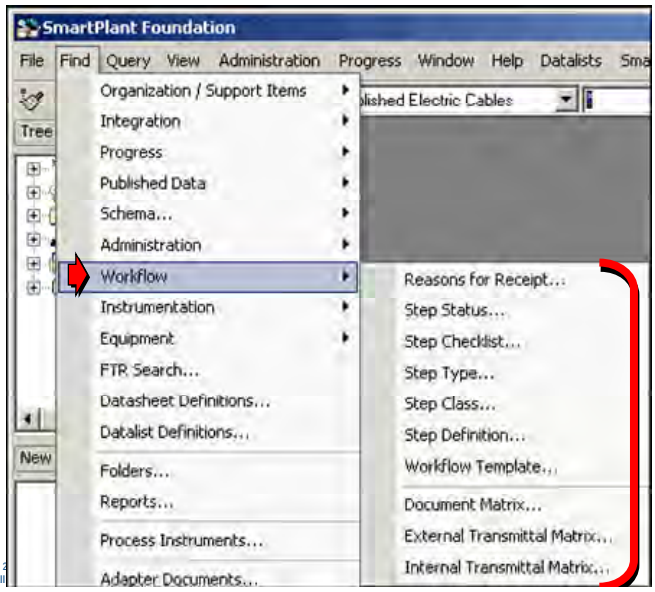
A workflow administrator creates workflow templates consisting of the workflow steps with defined recipients, which are *users*, *roles*, or *matrices*. These templates are conditionally made available to manage different classes of object. Shortcut menu commands on objects in the client are used to send them down a workflow and then manage the item in that workflow. All workflow communication to the SmartPlant Foundation user is performed in the **To Do List** and optionally by a regular e-mail.

When a workflow step requires action from a user, that step is presented in the user's *To Do List* where the appropriate commands are presented on the shortcut menu. The appearance of the action in the user's *To Do List* clearly shows if it is overdue.

7.1 Creating Workflows

Using the **SmartPlant Foundation Workflow** commands from the Desktop client, you can create, view, update, copy, and terminate a workflow.

SPF Workflow Administration



- **To perform Workflow administration operations, select the *Find > Workflow* commands.**

The *Workflow Template* defines the set of steps through which the object in a workflow must pass. When an object is attached to a workflow, the workflow template definition is copied and used to manage that item. The copied workflow (referred to as the instantiated workflow) is used to record all step completion dates, user comments, etc. With valid permissions, a user can modify some of the details and recipients of steps that have not yet been started without changing the original template definition. The steps in the template workflow are selected from a library of step definitions to ensure a consistent look and feel to all workflows.

7.1.1 Step Types

Step Types define the internal behavior of steps and cannot be altered without significant customization of the code. Each step is assigned to a step classification, which contains a step type definition. The Step Type dictates how the step is managed by the system; the methods determine the functionality available to the user in the client. The default step types delivered with the Workflow model are:

Step Types

Step Types define the behavior of steps and step classifications

The default step types are:

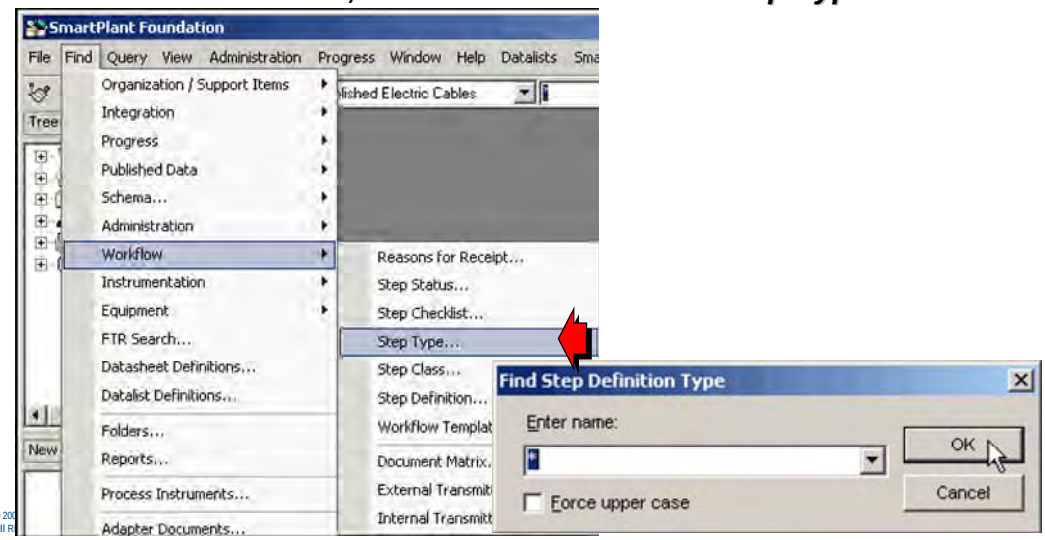
Step	Type	Description Purpose
I	Information	This step contains no subsequent steps.
W	Working	This step assigns work to the steps recipients.
P	Process	This step is a non-interactive step.
V	Voting	This step is for processing the results of parallel steps.
S	Sign Off	This step is for document sign off.

© 2005. Intergraph Corp.
All Rights Reserved.

You can also view information about the step type. To view step type information, perform a search for the step type.

Step Types

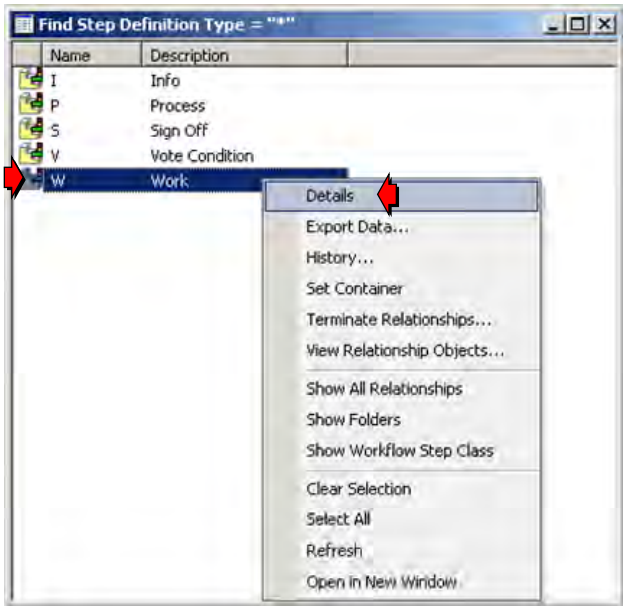
- **From the menu, click *Find > Workflow > Step Type***



The step types will be displayed in a list view browser window.

Step Types

- **Right-click on the W step type and click *Details* from the dynamic menu.**

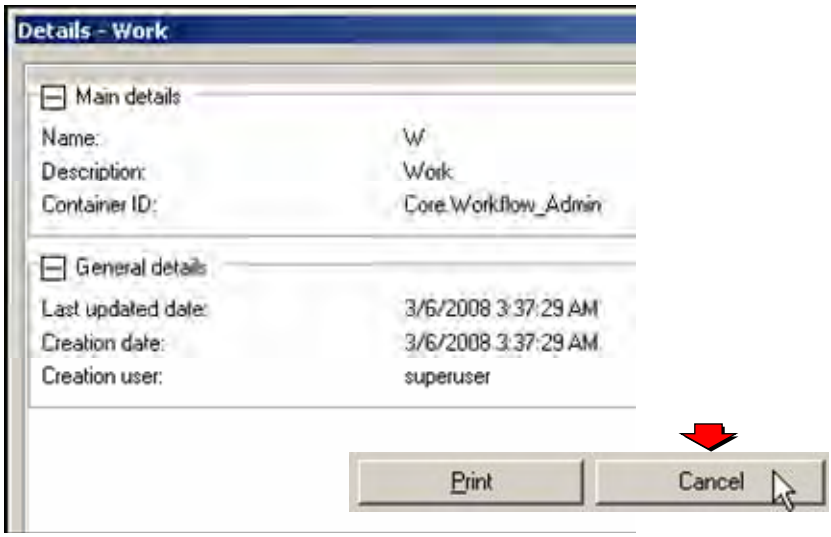


© 2005. Intergraph Corp.
All Rights Reserved.

Selecting the **Details** command displays the detail information for the selected step type as shown below.

Step Types

The *Step Type Details* form is displayed.



© 2005. Intergraph Corp.
All Rights Reserved.

7.1.2 Step Classes

Step Classes are used in conjunction with step types to define the behavior and appearance of steps. Methods are added to each step class to define commands that can be performed when a particular step is located in the user's inbox. The default step classes delivered with the Workflow module are:

Step Classes

Step Classes are used in conjunction with step types to define the behavior and appearance of steps

The default step classes are:

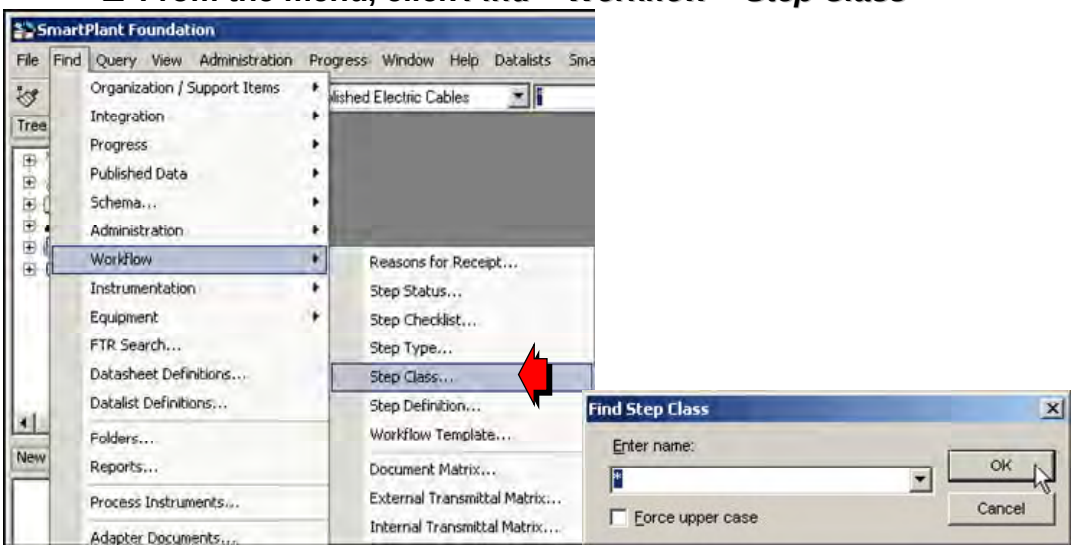
Step Class	Description	Step Type
Info	For Information	I
Assignment	Work Assignment	W
Approval	Approval Voting	W
Voting	Voting Result Testing	V
Sign Off	Automatic Document Revision Sign Off	S
E-mail Owner	E-mail Owner of the Change Object	P

© 2005. Intergraph Corp.
All Rights Reserved.

Step classes organize steps with similar functionality.

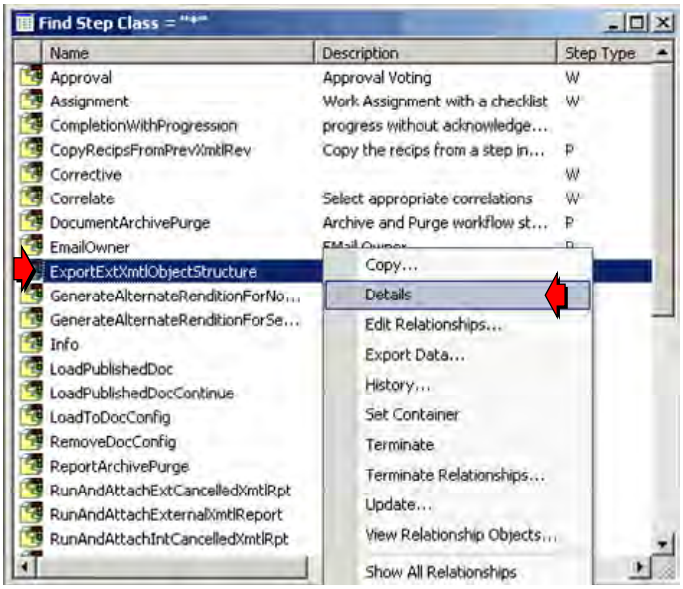
Step Classes

- **From the menu, click *Find > Workflow > Step Class***



To view step class information, right-click the step class that you want and choose **Details**.

Step Classes

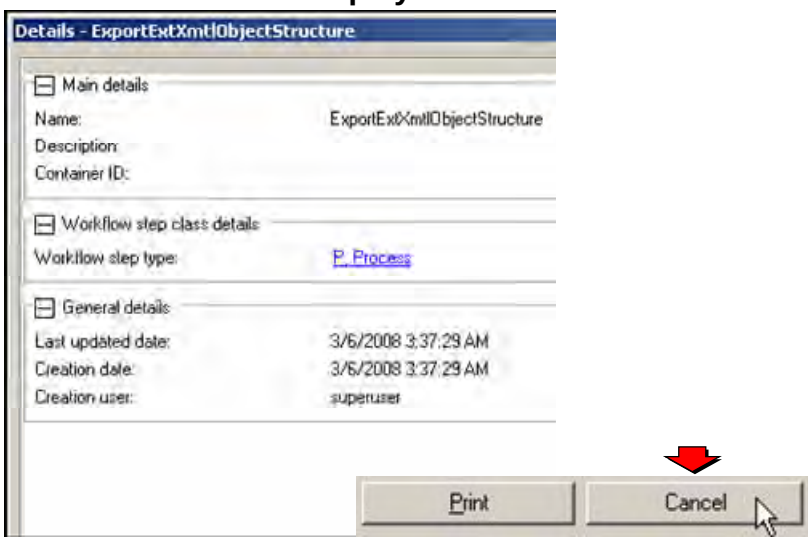


- **Right-click on the ExportExtXmtlObject Structure step class and click *Details* from the dynamic menu.**

Selecting the **Details** command displays the *Step Class Details* form for the selected step class as shown below.

Step Classes

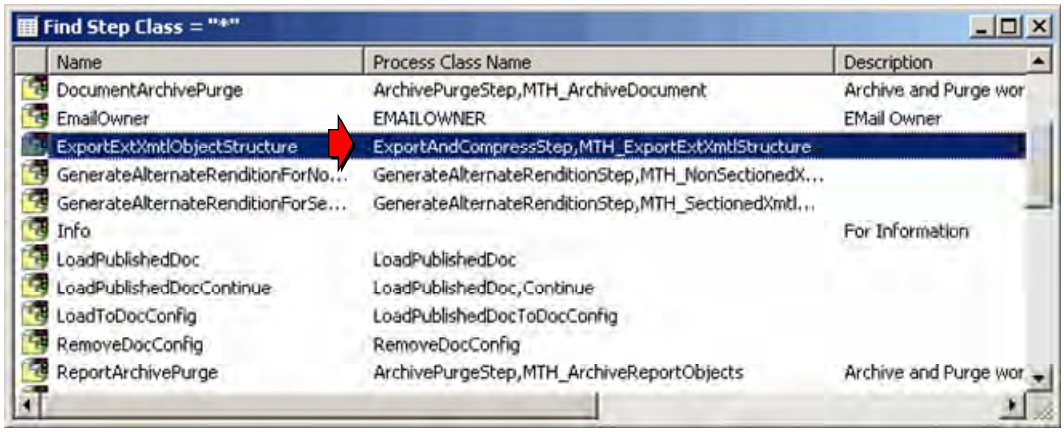
The *Step Class Details* form is displayed.



© 2005. Intergraph Corp.
All Rights Reserved.

Step Classes

In this example, this process step is used to run a predetermined process, ExportExtXmtlStructure.



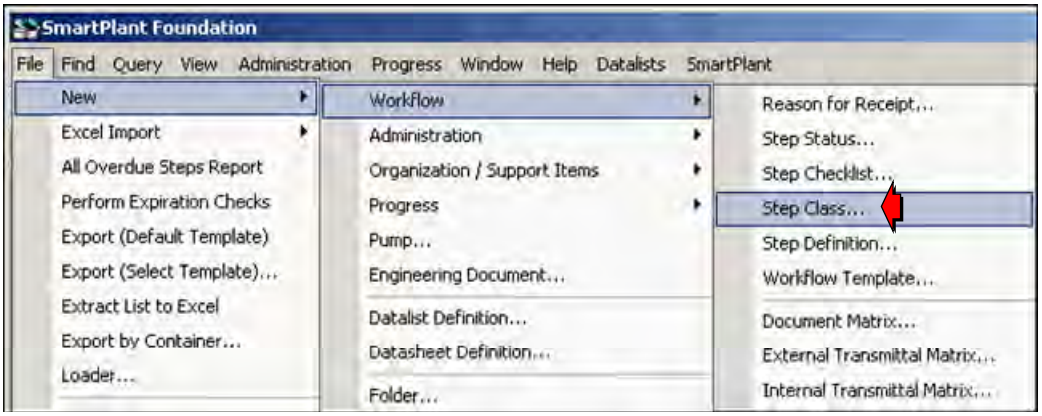
© 2005. Intergraph Corp.
All Rights Reserved.

7.1.3 Create a Step Class

A workflow can be customized by adding new site specific **Step Classes**.

Step Classes

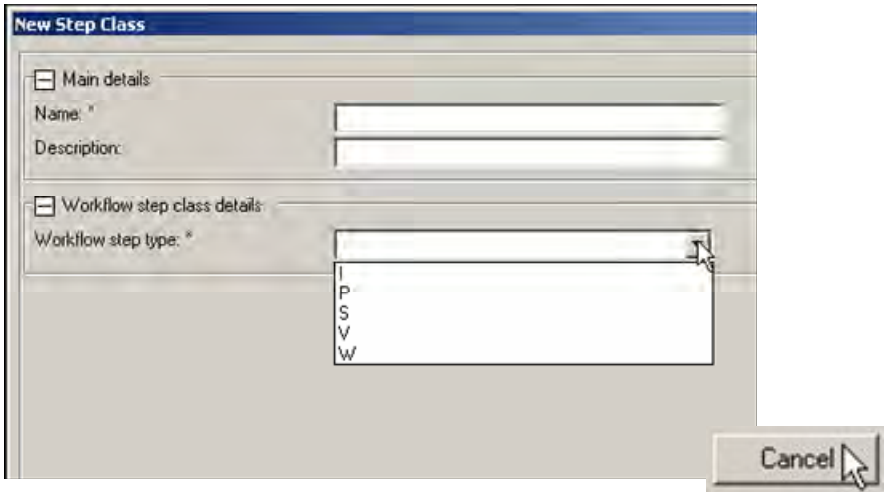
- **Select *File > New > Workflow > Step Class* from the menu.**



© 2005. Intergraph Corp.
All Rights Reserved.

Step Classes

The *New Step Class* form is displayed.



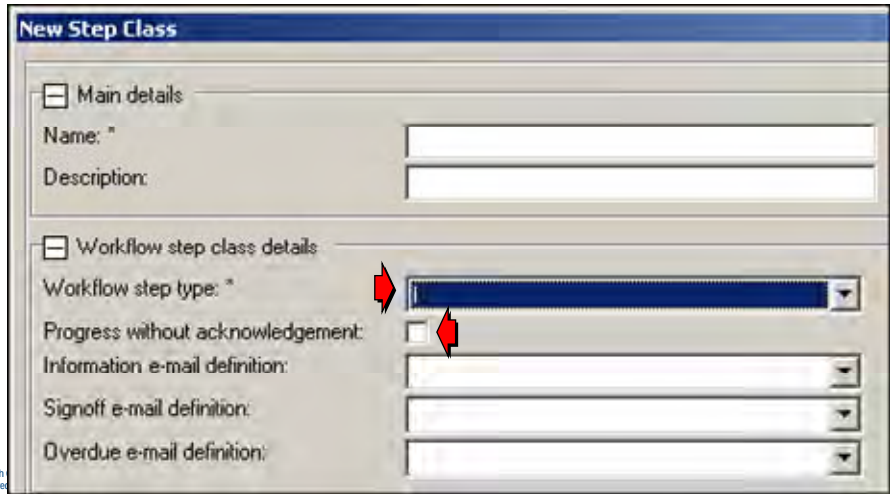
© 2005. Intergraph Corp.
All Rights Reserved.

1. In the *New Step Class* dialog box, type the name of the step class in the **Name** box.
2. In the **Description** box, type a description relevant to your step class.

3. Select the **Workflow step type**.

Step Classes

- **If you select I (Information step), you can also select the Progress without acknowledgement check box.**



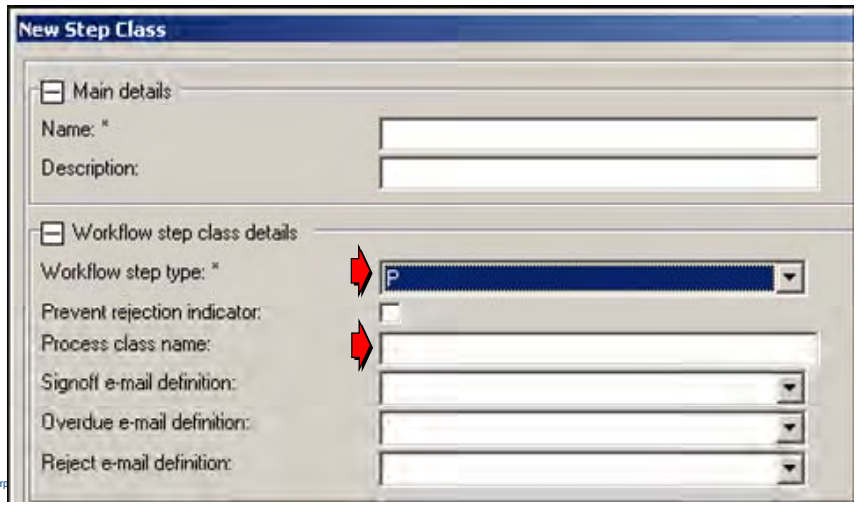
© 2005. Intergraph
All Rights Reserve

The following fields are used on the *New Step Class* form:

- **Progress without acknowledgement** - To allow the workflow to continue, but leave the step in the user's inbox (waiting for the user to acknowledge its receipt). This option is only available if you select the step type of information (I).
- **Information e-mail definition** - Defines the e-mail to be sent to the recipient when an information step is assigned.
- **Signoff e-mail definition** - Defines the e-mail to be sent to the recipient when a step sign off is required.
- **Overdue e-mail definition** - Defines the e-mail to be sent to the recipient when step action is overdue.

Step Classes

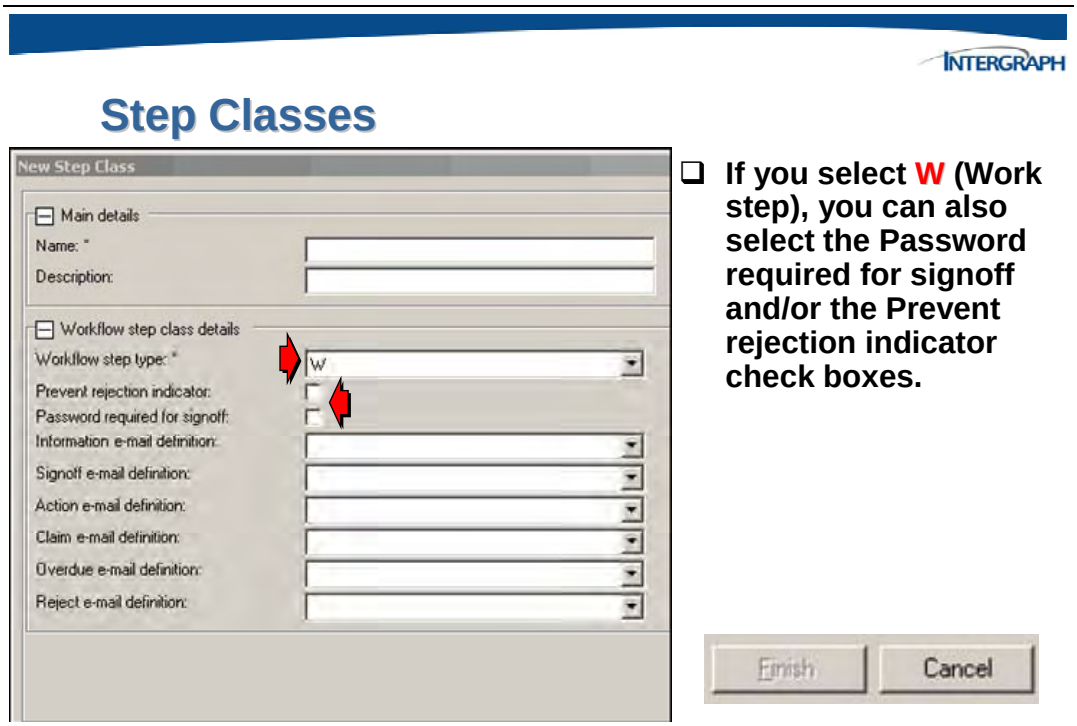
- **If you select P (Process step), you must also type a name in the Process class name box.**



© 2005. Intergraph Corp
All Rights Reserved.

New Step Class form fields con't:

- **Prevent rejection indicator** - Selecting the Prevent rejection indicator check box will prevent the workflow step from being rejected to another workflow.
- **Process class name** - This step can be used to set a value on an attribute of the object progressing down the workflow.
- **Reject e-mail definition** - Defines the e-mail to be sent to the recipient when a step is rejected.



New Step Class form fields con't:

- **Password required for signoff** - Select to request the user to enter a password to sign off the step in the workflow.
- **Action e-mail definition -** Defines the e-mail to be sent to the recipient when an action step is required.
- **Claim e-mail definition -** Defines the e-mail to be sent to the recipient when step action has been claimed.

4. Click **Finish**.

7.1.4 Step Definitions

Steps are used during the definition of a workflow template. Each step in a workflow defines a process that must be performed in order to complete the workflow. Step Types and Step Classes are used to define the behavior of each step. Every step must be related to a step definition, which is used to describe the behavior of the step.

Using the commands in the **Workflow** model, you can create, update, and delete step definitions for a workflow.

Step Definitions

A Step Definition defines a process that must be performed in order to complete the workflow

A Step is made up of:

- **Name**
- **Description**
- **Step Class**
- **Post Class Name**

Post class name format:

<Process class name>.<class>.<method>,parameter1,parameter2……..

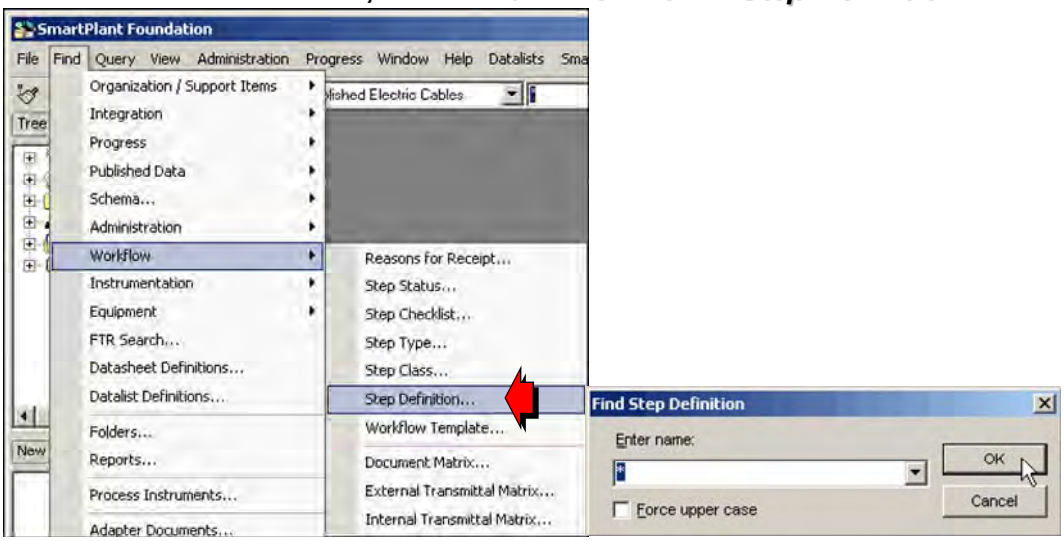
CREATECHKLISTFORSTEP,XmtlSectionDocs,True

© 2005. Intergraph Corp.
All Rights Reserved.

You can also view information about the step.

Step Definitions

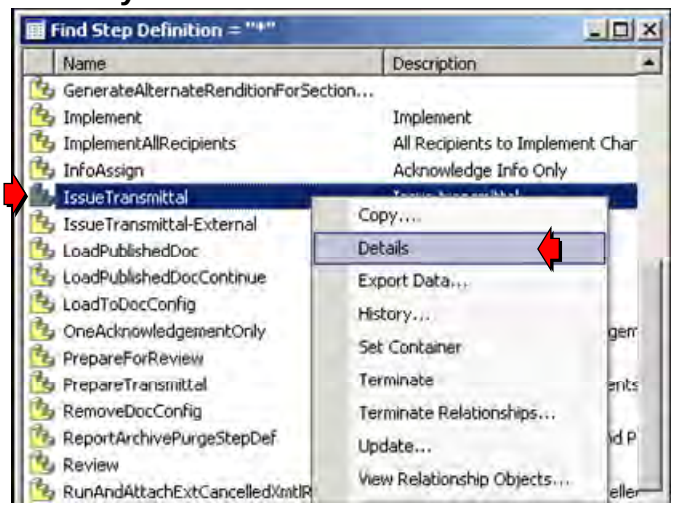
- **From the menu, click *Find > Workflow > Step Definition*.**



To view step information, or to see which workflows a step is assigned to, right-click the step and choose **Details** or **View Relationship Objects**.

Step Definitions

- **Right-click on the IssueTransmittal step and click *Details* from the dynamic menu.**

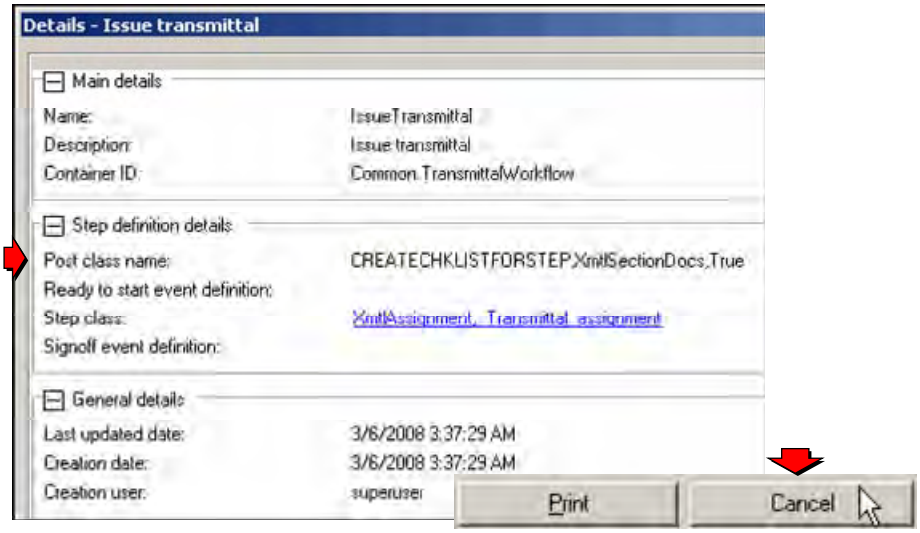


© 2005. Intergraph Corp.
All Rights Reserved.

Selecting the **Details** command displays the *Step Details* form for the selected step definition as shown below.

Step Definitions

The *Step Details* form is displayed.

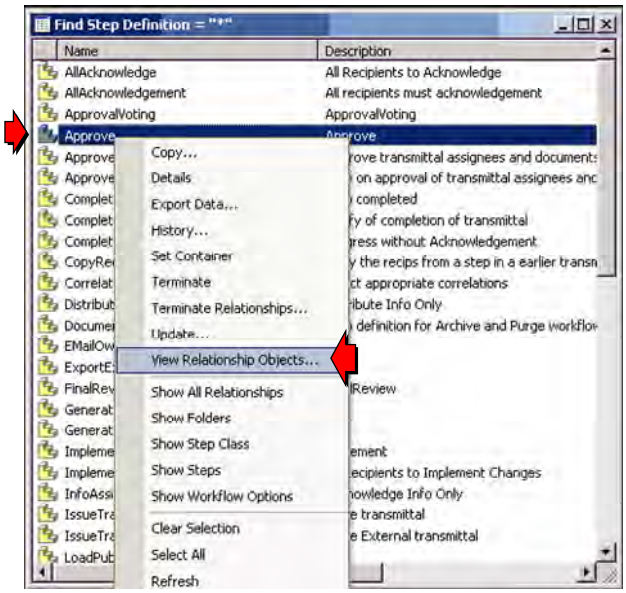


© 2005. Intergraph Corp.
All Rights Reserved.

Selecting the **View Relationship Objects** command displays the Workflows where the Step Definition is used as shown below.

Step Definitions

- **Right-click the Approve step and click *View Relationship Objects* from the dynamic menu.**

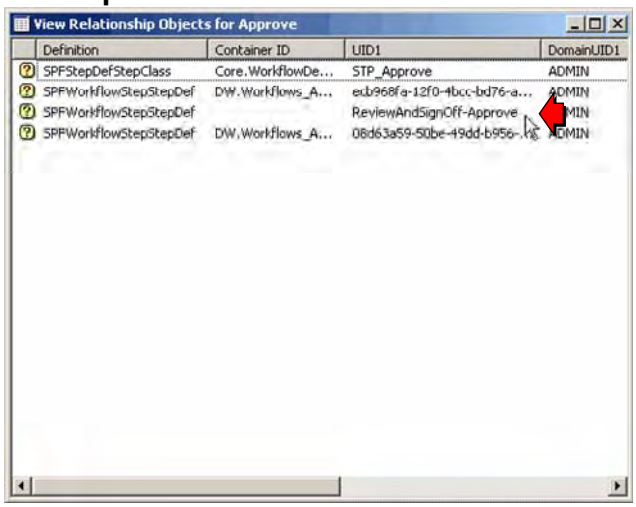


© 2005. Intergraph Corp.
All Rights Reserved.

This will display which workflows are currently using the *Approve* step.

Step Definitions

- **The Approve step displays in a browser to show the relationship to workflows.**



© 2005. Intergraph Corp.
All Rights Reserved.

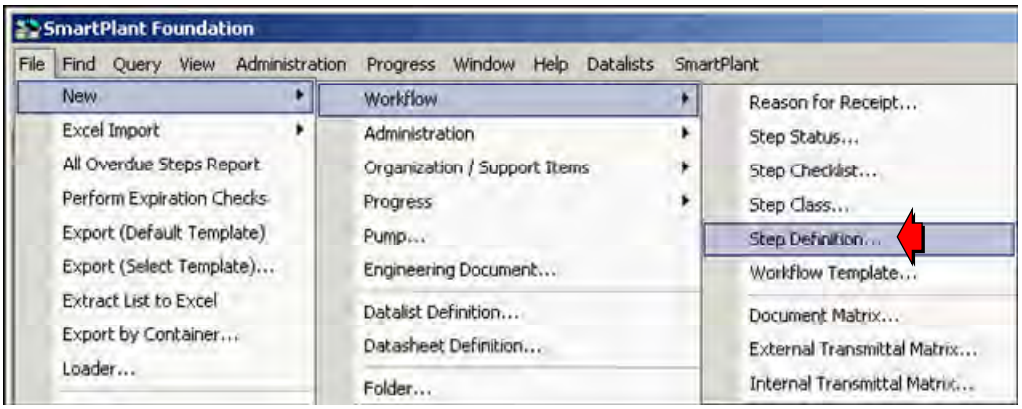
7.1.5 Create a Step Definition

Customizing a workflow can include creating new workflow step definitions.

INTERGRAPH

Step Definitions

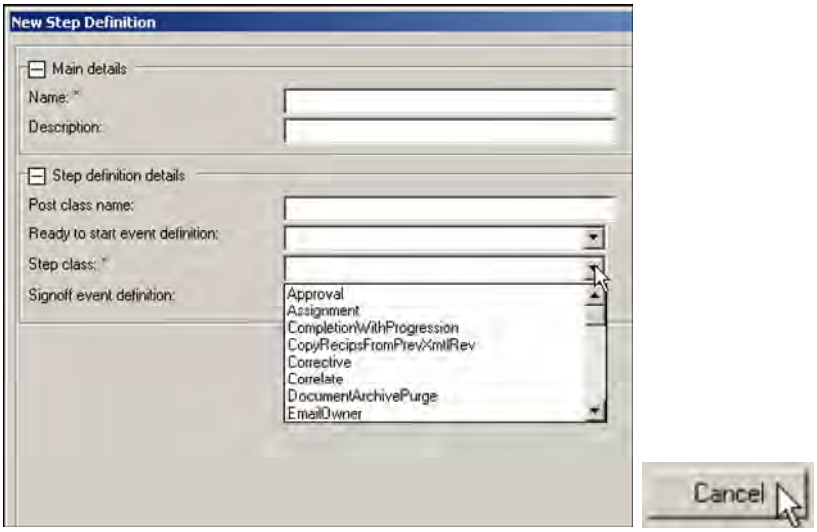
- **Select *File > New > Workflow > Step Definition* from the menu.**



© 2005. Intergraph Corp.
All Rights Reserved.

Step Definitions

The *New Step Definition* form is displayed.



© 2005. Intergraph Corp.
All Rights Reserved.

1. Type a unique *name* and *description* for the step definition.
2. In the **Step class** box, select the step class to which the definition is related.
3. In the **Post class name** box, type the post class name.

   The post class name is the name of the process class to be run after the step related to this definition completes.

   The post class name format is:

   <Process class name>.<class>.<method>,parameter1,parameter2, and so on

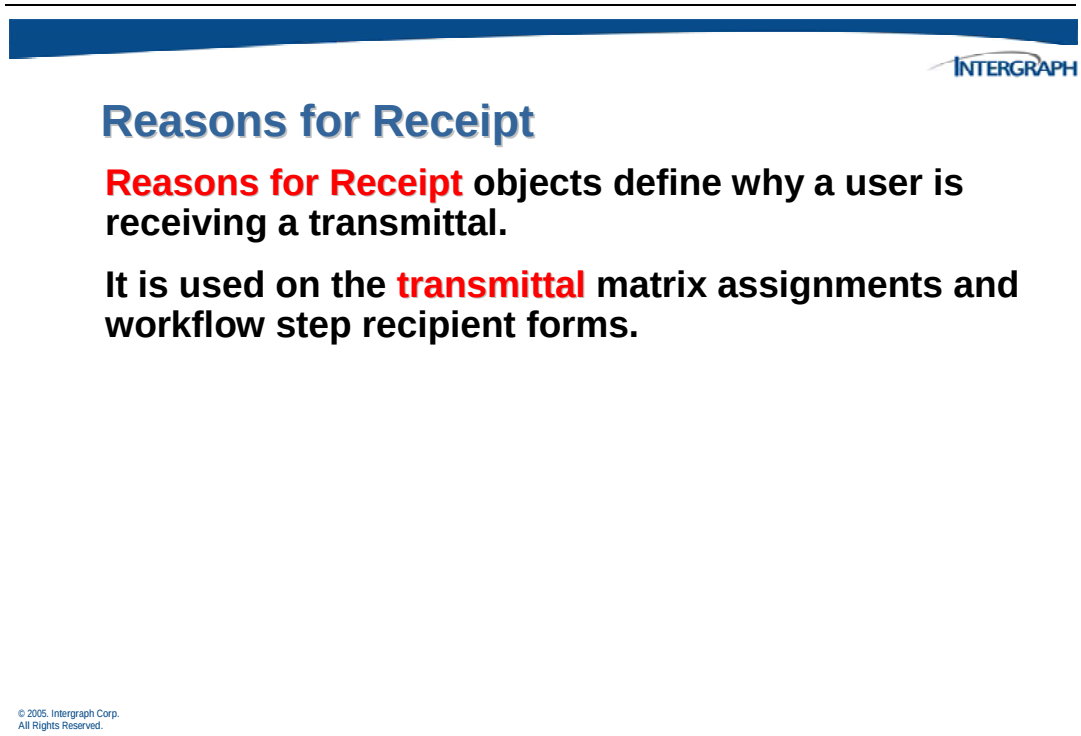
   where the number of parameters is determined by the method.

   Example: CREATECHKLISTFORSTEP,XmtlDocs,True

   The post class name provides a hook for customization, which can be used to perform additional processing after the step has completed and before the next step starts. For example, there is a post class name to generate a checklist based on objects related to the main object in the workflow and this is used on a transmittal issue step to create a checklist item for each document on the transmittal.
4. Click **Finish** to create the new step definition.

7.1.6 Using Reasons for Receipt

Reasons for receipt allow you to define why a user is receiving an assignment, such as work, information only, and comment. This object is used on the transmittal Assign Recipients and workflow step recipient forms. The **Reason for Receipt** object contains the following properties: *Name*, *Description* and *Information Only* indicator.

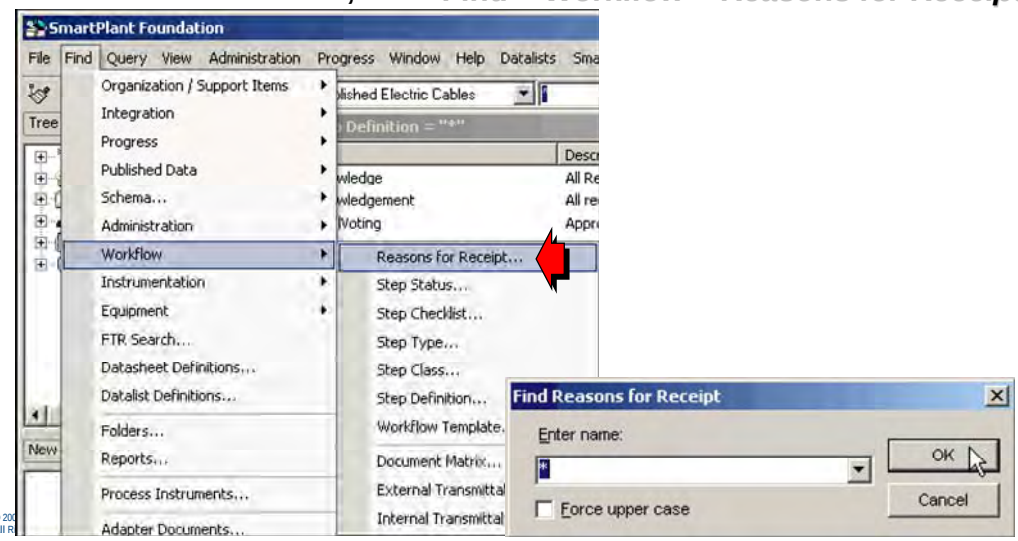


Using the **Workflow** model, you can create, update, and delete reasons for receipt. You can also view information about existing reasons for receipt.

To view reason for receipt information, search for the reason for receipt objects you want.

Reasons for Receipt

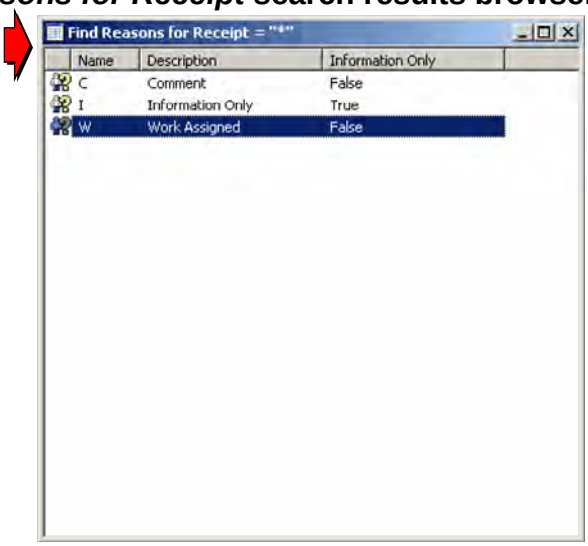
- **From the menu, click *Find > Workflow > Reasons for Receipt***



The search will display the *Reason for Receipt Properties* results browser for the selected reason for receipt as shown below.

Reasons for Receipt

The *Reasons for Receipt* search results browser is displayed.



© 2005. Intergraph Corp.
All Rights Reserved.

7.1.7 Create a Reason For Receipt

Custom **Reasons for Receipt** can be created for use in a workflow. The following example demonstrates adding a custom *Reason for Receipt*.

Interactive Activity

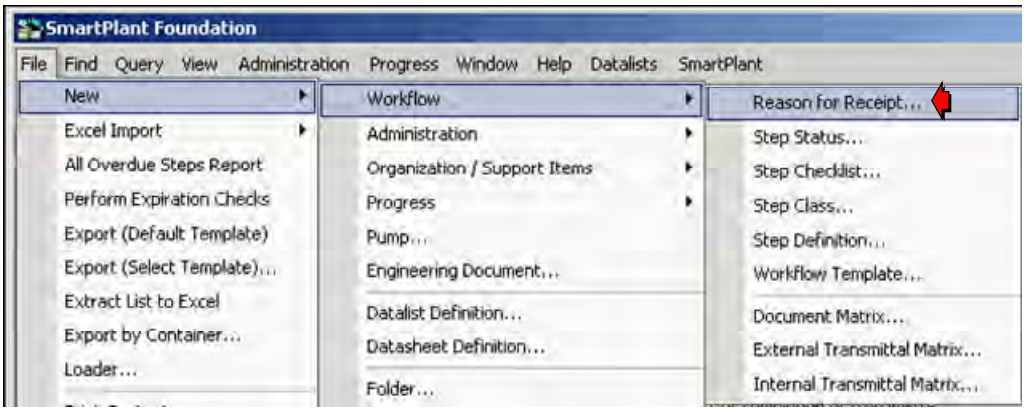
You can follow along interactively as we build a sample workflow.

Use the sample in this chapter as an example of building and using a workflow.

© 2005. Intergraph Corp.
All Rights Reserved.

Reasons for Receipt

- **Select *File > New > Workflow > Reasons for Receipt* from the menu.**

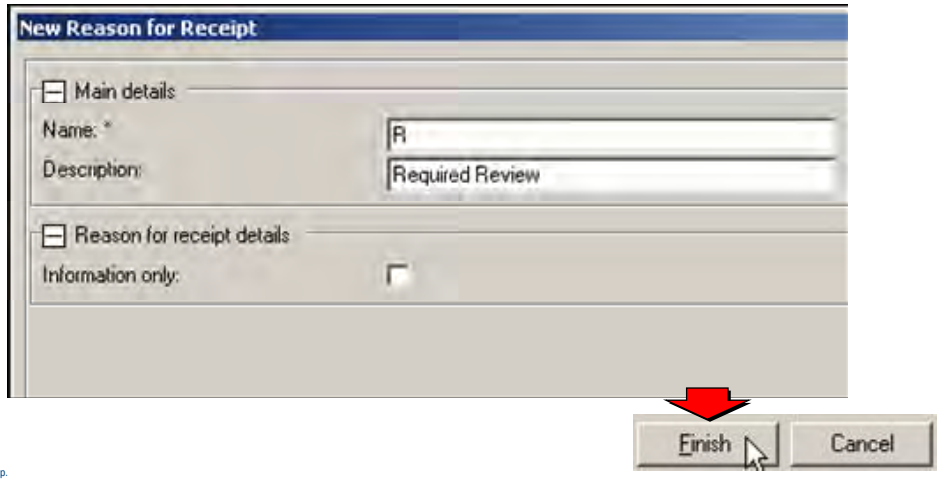


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Reason for Receipt* form will appear.

Reasons for Receipt

- **Enter *New Reason for Receipt* information and click Finish.**

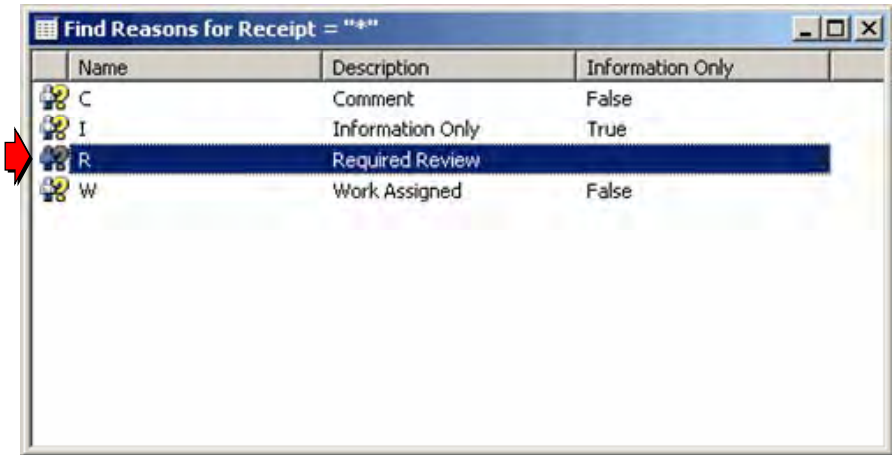


© 2005. Intergraph Corp.
All Rights Reserved.

1. In the *New Reason For Receipt* form, type a name and description in the **Name** and **Description** boxes.
2. If the reason for receipt will be for information only, select the **Information only** check box.
3. Click **Finish**.

Reasons for Receipt

Shown here are the results for the new *Reasons for Receipt*.



© 2005. Intergraph Corp.
All Rights Reserved.

7.1.8 Workflow Check Lists

Check Lists are a series of items, tasks, or questions that the user can be forced to complete before a step can be completed in a workflow. Each step in a workflow can have a checklist (to be completed by the recipients) attached to it. Completion of these check lists can be mandatory or optional. The checklist items are presented in the *To Do List* against the step. Checklists are defined with a **name** and **description**. Each item in the checklist is defined as a simple text string. Checklist objects are used when defining a workflow template but must exist before they can be referenced by the workflow.

Check Lists

Check list can be attached to a workflow step and it can be configured to be completed before the workflow step can be completed.

A check list can include several items:

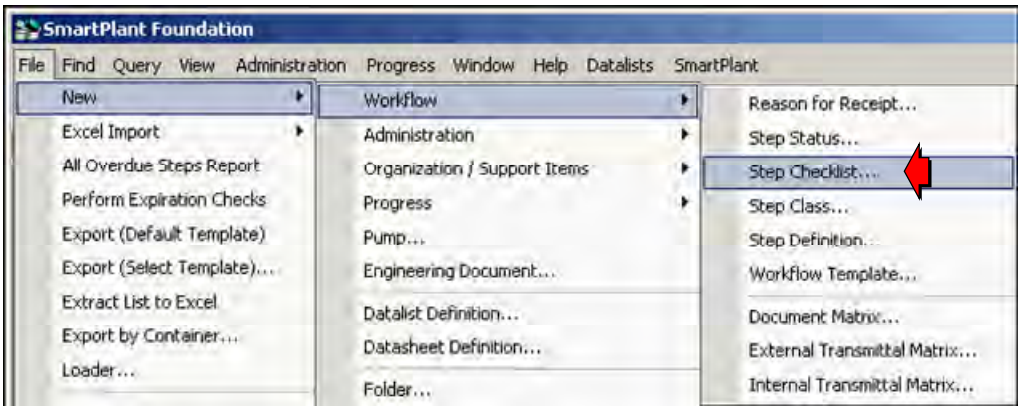
- **Check list name**
- **Check list description**
- **Check list items**

© 2005. Intergraph Corp.
All Rights Reserved.

To create a new workflow check list:

Creating Check Lists

- **Select *File > New > Workflow > Step Checklist* from the menu.**

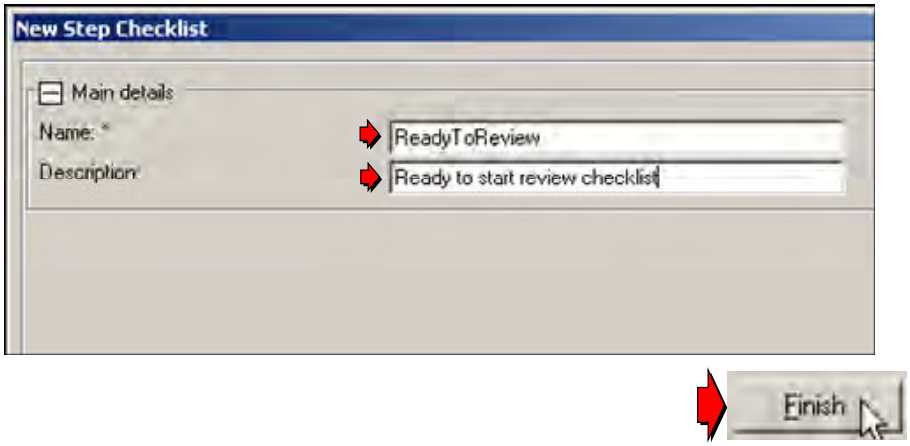


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Step Checklist* form will display.

Creating Check Lists

- **Specify a required Name*, optional Description and click Finish.**



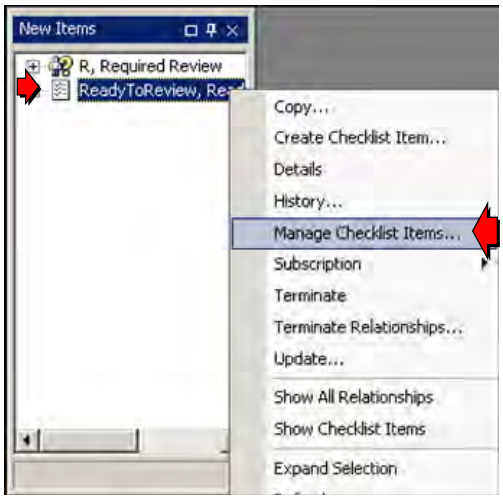
© 2005. Intergraph Corp.
All Rights Reserved.

The following fields are used to define a checklist:

- **Name -** a unique name to define the checklist
- **Description** - a brief description for the check list, which is an optional field

Creating Check Lists

- **Right-click the ReadyToReview Check List and click *Manage Checklist Items* from the dynamic menu.**

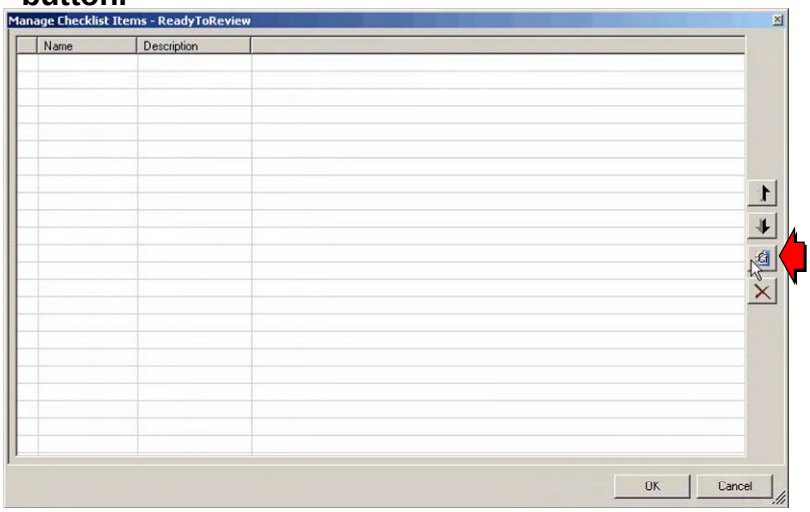


© 2005. Intergraph Corp.
All Rights Reserved.

The *Manage Checklist Items* dialog will appear.

Creating Check Lists

- **On the *Manage Checklist Items* dialog, click the create button.**

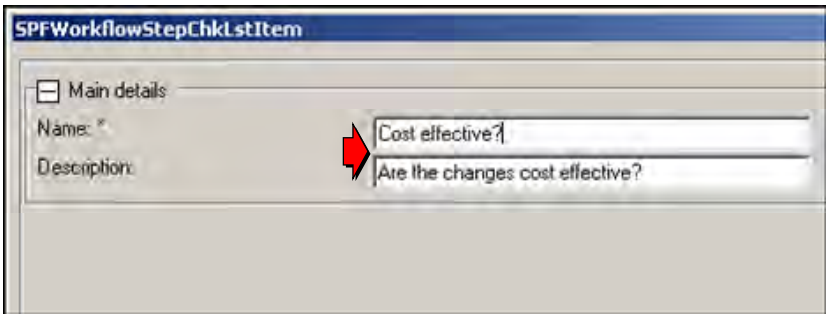


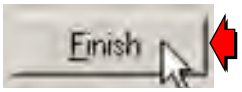
© 2005. Intergraph Corp.
All Rights Reserved.

The *SPFWorkflowStepChkLstItem* form will appear.

Creating Check Lists

- **Enter the values for the new *Checklist Item*.**



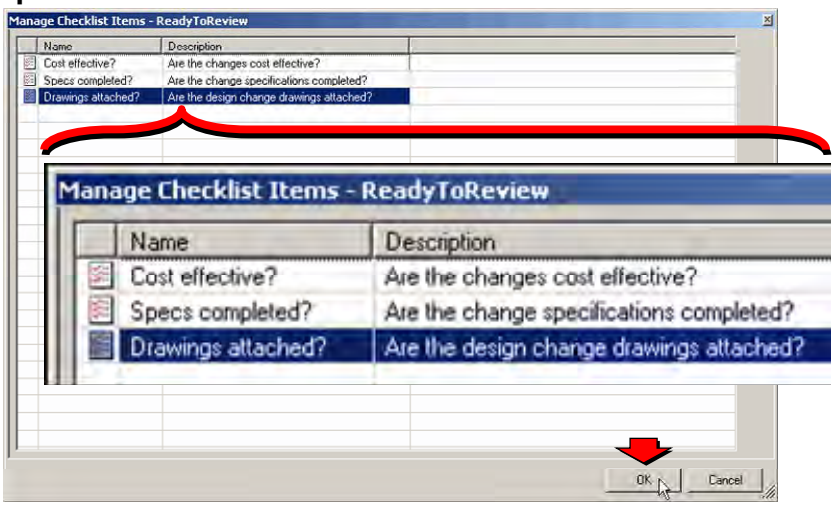


© 2005. Intergraph Corp.
All Rights Reserved.

Repeat the procedure to add the remaining *Checklist Items*.

Creating Check Lists

- **The new Checklist Items are added to the relationship pane.**

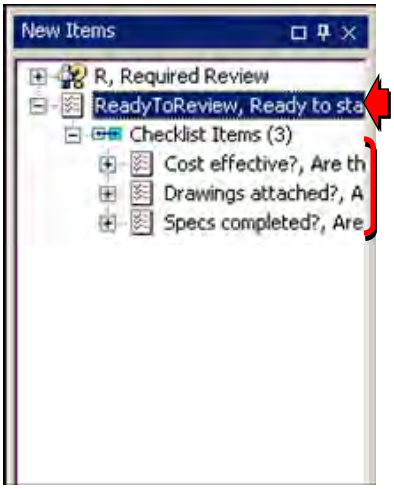


© 2005. Intergraph Corp.
All Rights Reserved.

The *Checklist* now relates to three *Checklist Items* as shown below.

Creating Check Lists

- **Expand the *Checklist* to see the related *Checklist Items*.**

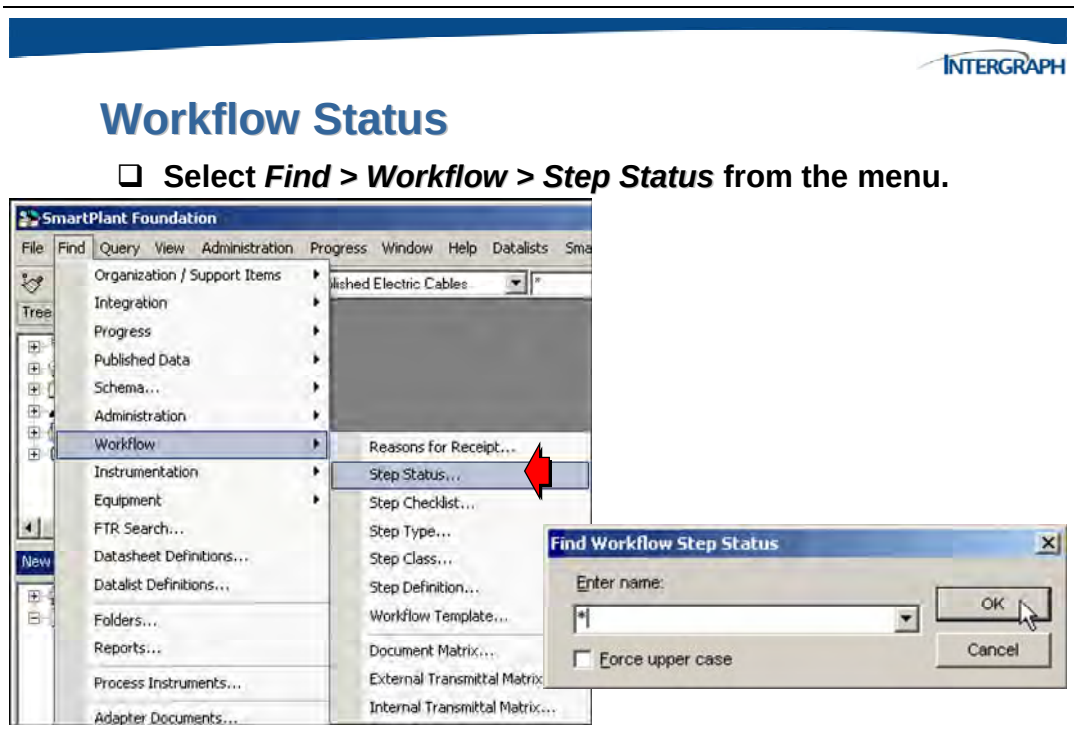


© 2005. Intergraph Corp.
All Rights Reserved.

7.1.9 Workflow Statuses

Status objects are used during the definition of a workflow template. A workflow status is displayed on object in a workflow to indicate how it is progressing through that workflow. The status is set on the object in the workflow at the completion of a step. This is configured on the workflow administration form.

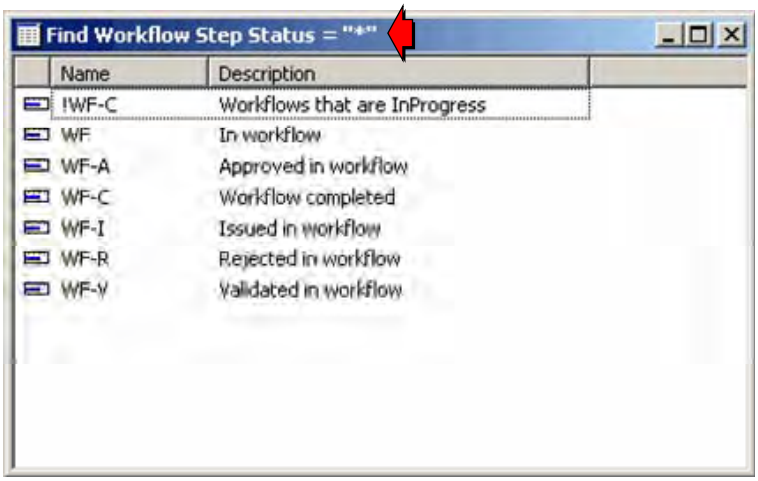
Perform a search to view the current statuses available to Workflow.



The search will display the *Step Status* results browser as shown below.

Workflow Status

The *Step Status* search results browser is displayed.

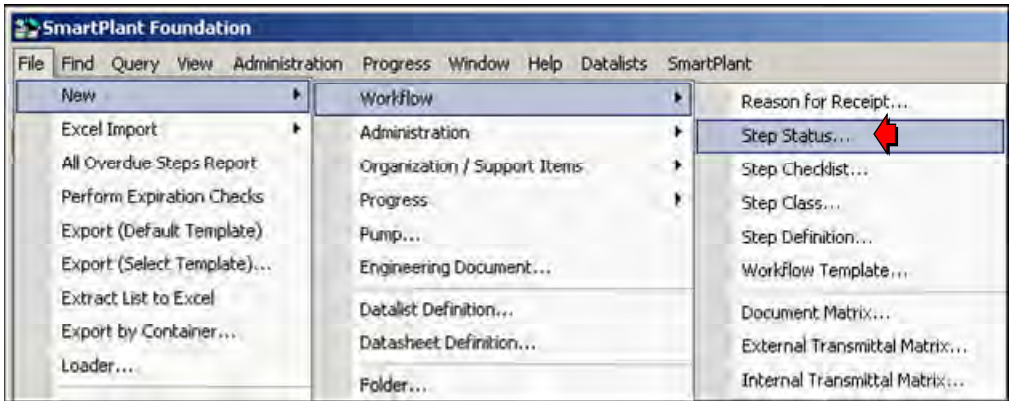


© 2005. Intergraph Corp.
All Rights Reserved.

The following example demonstrates how to add new statuses to Workflow.

Adding New Statuses

- **Select *File > New > Workflow > Step Status* from the menu.**

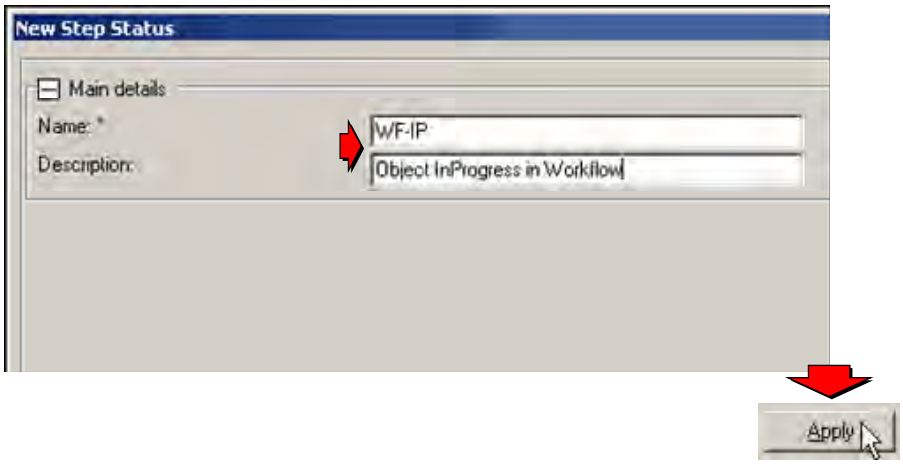


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Step Status* dialog will display.

Adding New Statuses

- **Enter a required status Name and an optional Description and click Apply.**

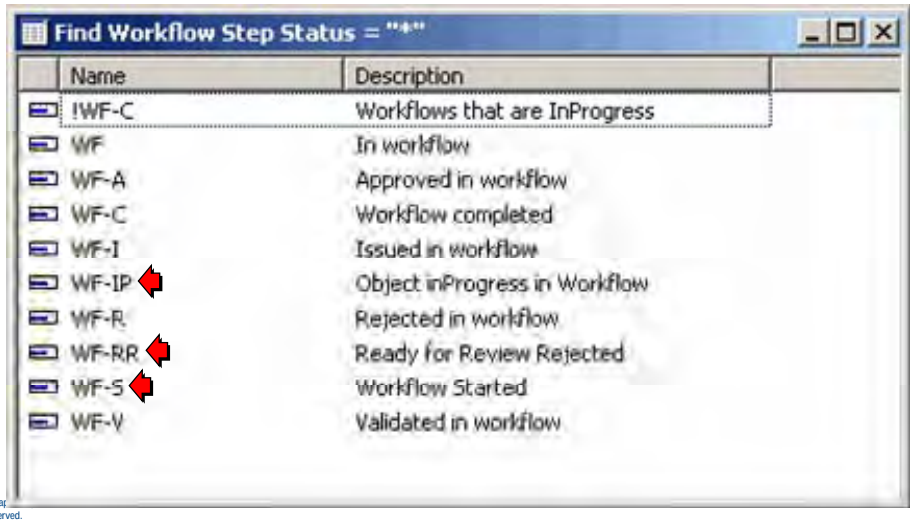


© 2005. Intergraph Corp.
All Rights Reserved.

Continue until all of the new statuses have been added, then click **Finish**.

Adding New Statuses

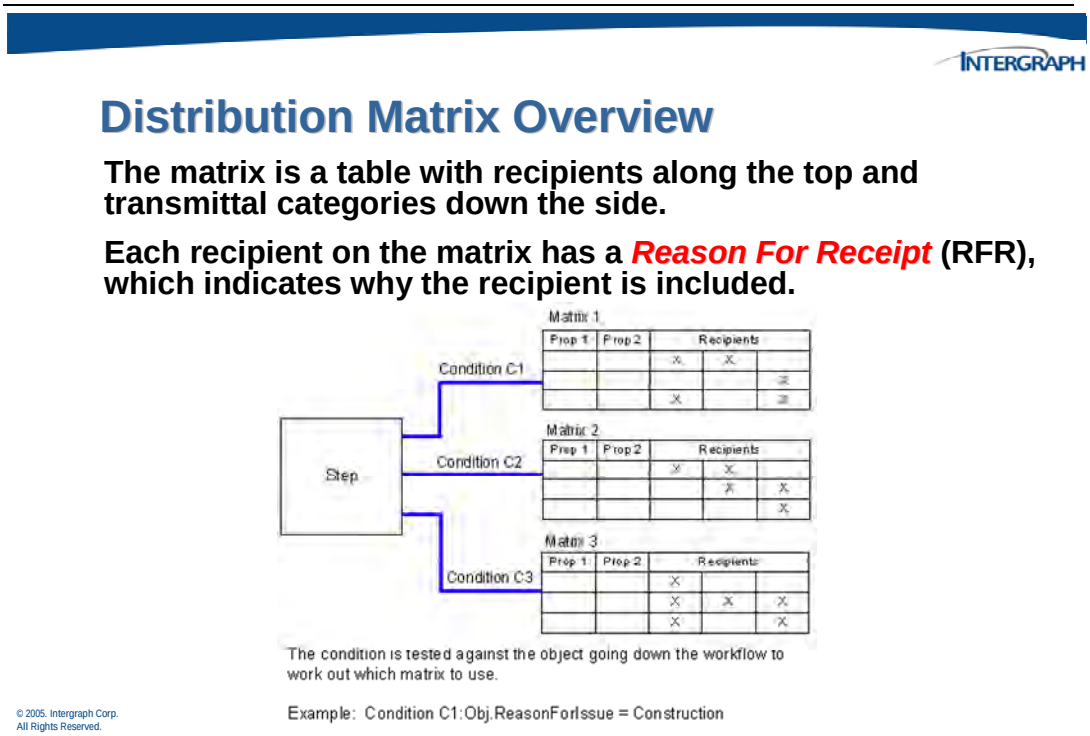
The new statuses are now available to be used in workflows



© 2005. Intergra
All Rights Reserved.

7.2 Distribution Matrix Overview

Distribution matrices control sets of recipients for a workflow step dependent on the properties and/or the relationships of the object in the workflow.



For example, transmittal matrices are created for a reason for issue and related to the step by a condition such as IsIssuedForConstruction.

Distribution Matrix Overview

The matrix categories can be based on any property or relationship on the object going down the workflow, such as document type, project, and contract.

Each cell in the table is marked with the reason for receipt for each recipient.

Distribution matrices determine which users receive the workflow actions and what tasks are required of those recipients. Separate matrix classes have to be defined for each object that has different mappings.

© 2005. Intergraph Corp.
All Rights Reserved.

Distribution Matrix Overview

Alternative matrices can be configured on a step and controlled by a condition.

To determine the recipients that are to be obtained from the matrix, the row properties are matched to extract the correct row from the matrix.

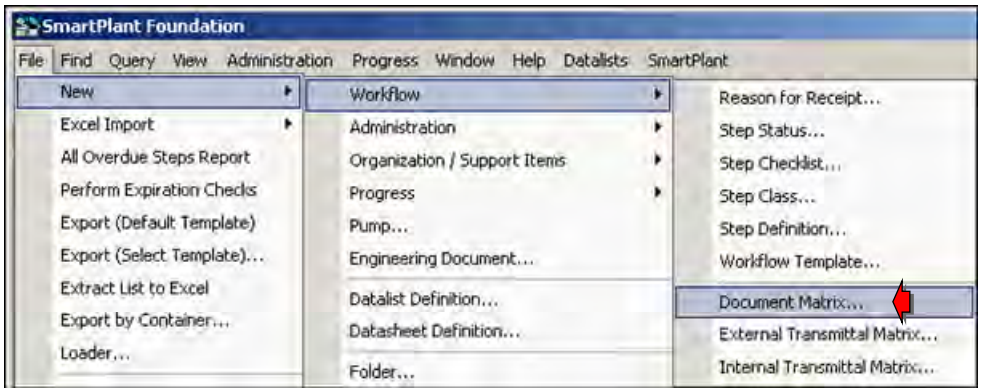
© 2005. Intergraph Corp.
All Rights Reserved.

7.2.1 Creating a Document Matrix

The following section will demonstrate how to create a **Document Matrix**.

Creating a Document Matrix

- **Select *File > New > Workflow > Document Matrix* from the menu.**

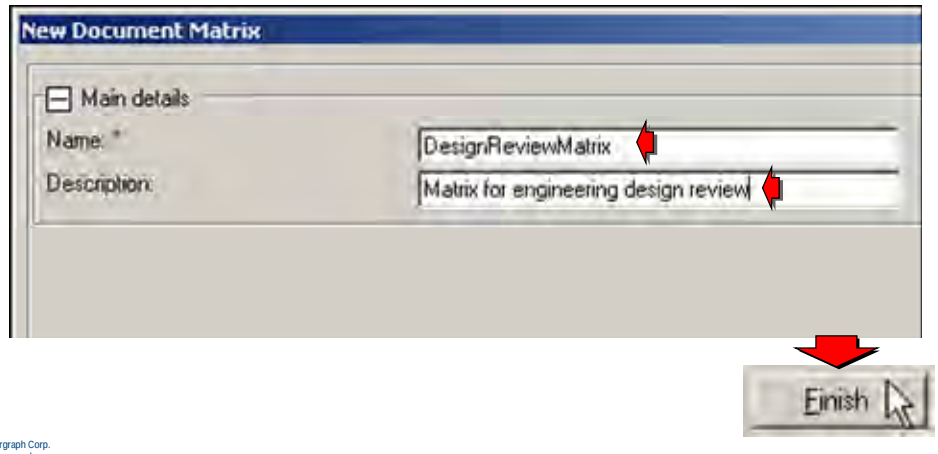


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Document Matrix* form will display.

Creating a Document Matrix

- **Enter the *Name* and optional *Description* for the Document Matrix.**

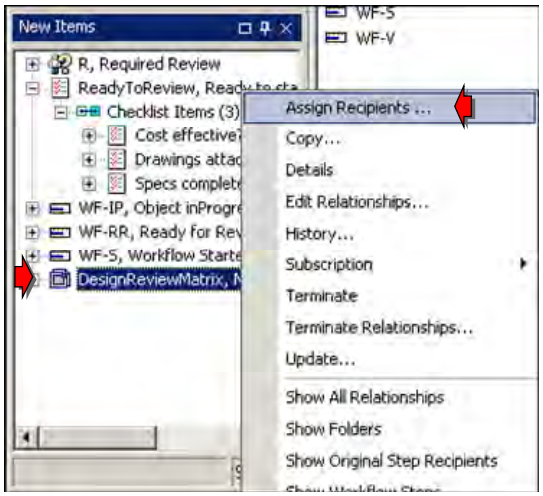


© 2005. Intergraph Corp.
All Rights Reserved.

Once the *Document Matrix* has been created, the next step is to assign the users that will be associated with this matrix.

Creating a Document Matrix

- **Right-click the Document Matrix and select *Assign Recipients* from the menu.**

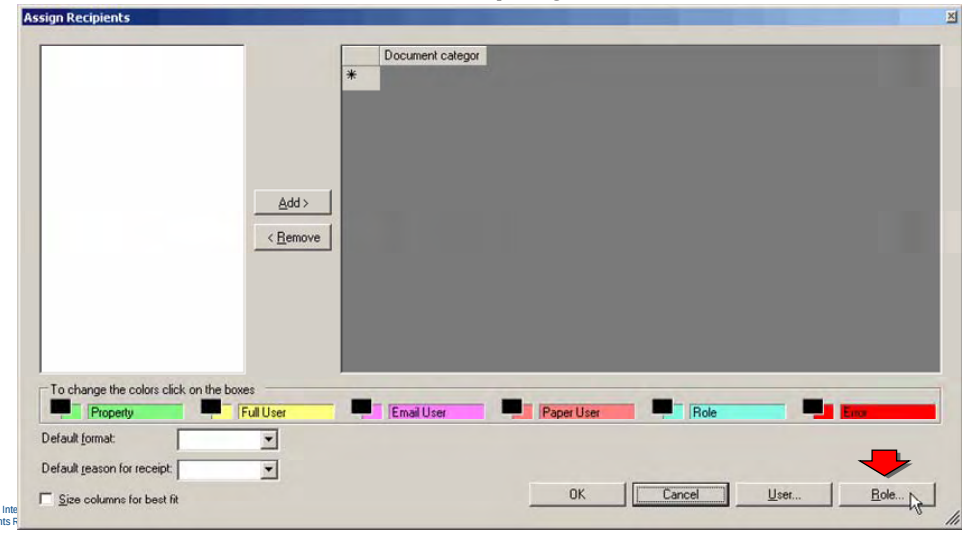


© 2005. Intergraph Corp.
All Rights Reserved.

The *Assign Recipients* form is used to specify the users or roles included in the selected distribution matrix, as well as the action required from each user or user group upon receipt of a transmittal.

Creating a Document Matrix

- **Select the Role button to query for available roles.**

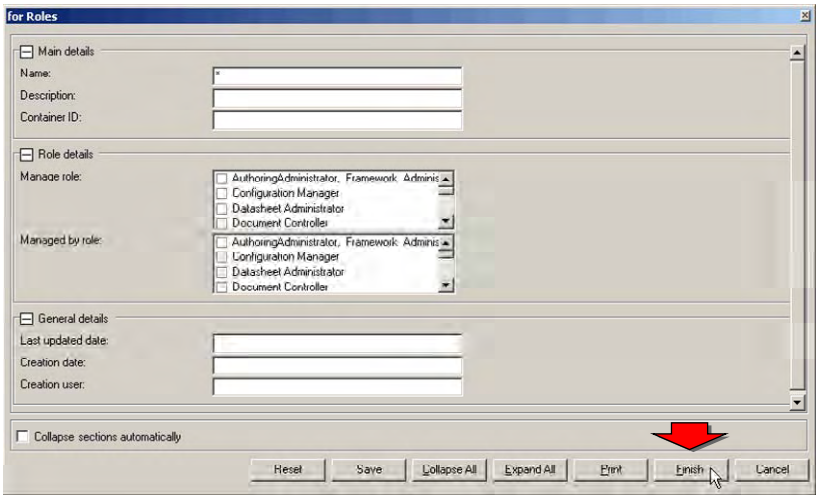


© 2005. Inte
All Rights R

A *Search for Roles* form will appear.

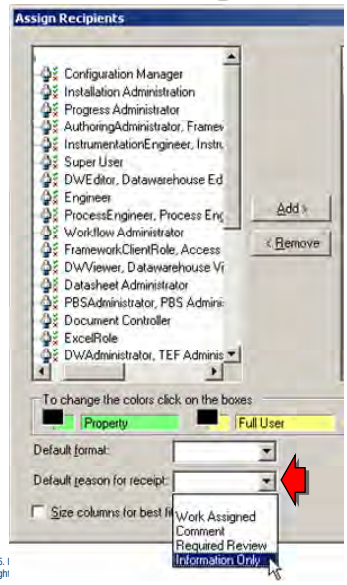
Creating a Document Matrix

- **Enter the search criteria to locate users you want to assign to the Document Matrix.**



© 2005. Intergraph Corp.
All Rights Reserved.

Creating a Document Matrix

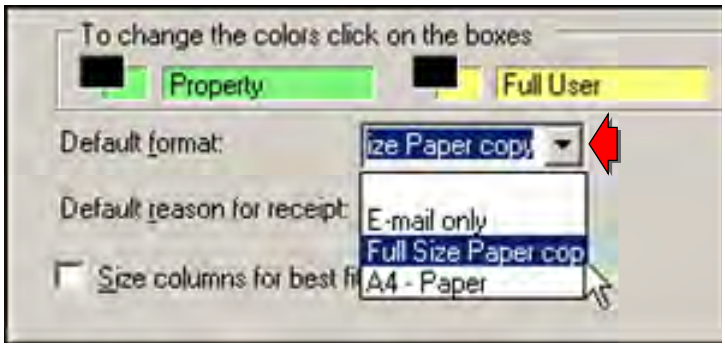


- **Set the *Default reason for receipt* for the Document Matrix.**

On the *Assign Recipients* dialog, there is a **Default format** and **Default reason for receipt**. Select the values that will be default settings for the Document Matrix.

Creating a Document Matrix

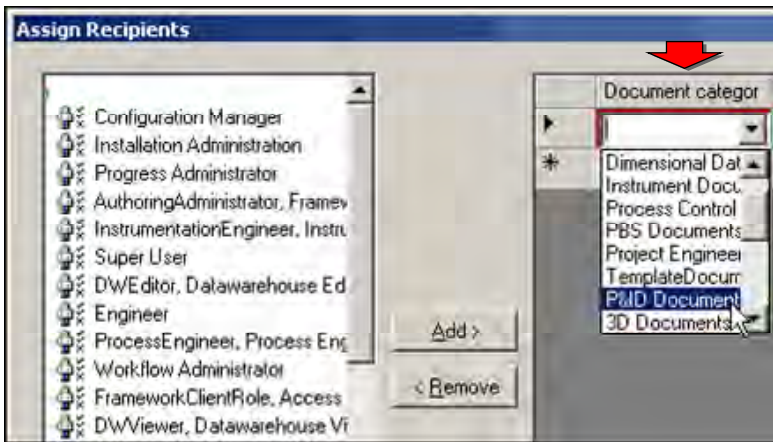
- **Set the *Default format* for the Document Matrix.**



© 2005. Intergraph Corp.
All Rights Reserved.

Creating a Document Matrix

- **Click in the *Document category* field to display a picklist of values.**

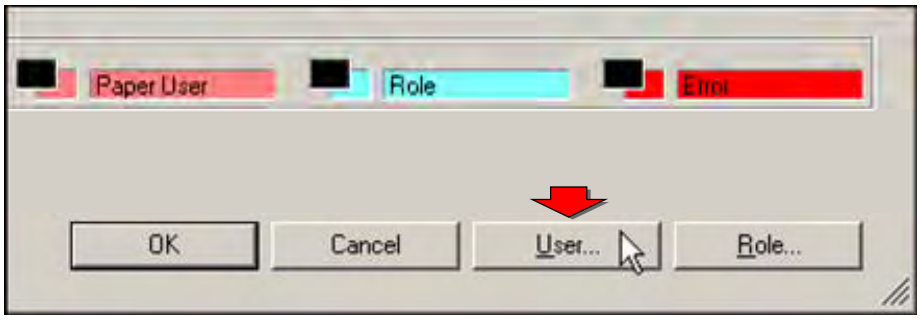


© 2005. Intergraph Corp.
All Rights Reserved.

Users and Roles can be added to the Document Matrix. After setting the *Reason for Receipt* and *Format* for each recipient, these settings appear in the matrix. The Document Matrix information to use when a workflow is used is determined by *Document category*.

Creating a Document Matrix

- **Select the User button to query for available users.**

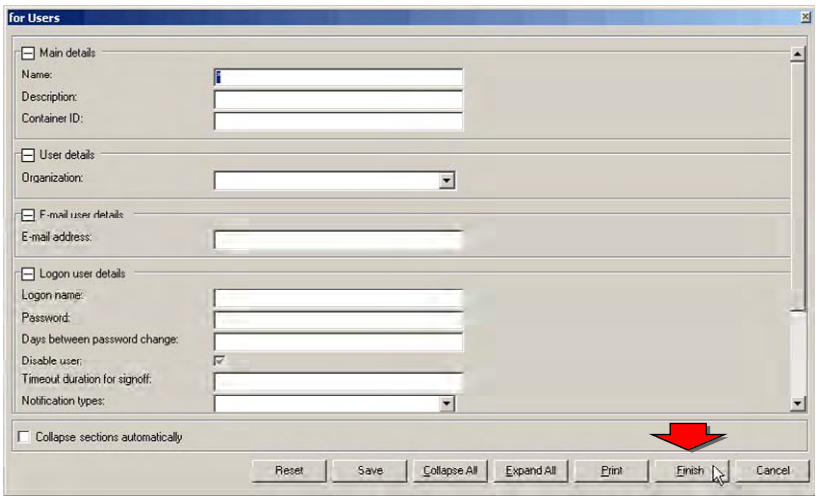


© 2005. Intergraph Corp.
All Rights Reserved.

Use the **User** button to populate the recipient list (left pane).

Creating a Document Matrix

- **Enter the search criteria to locate users you want to assign to the Document Matrix.**

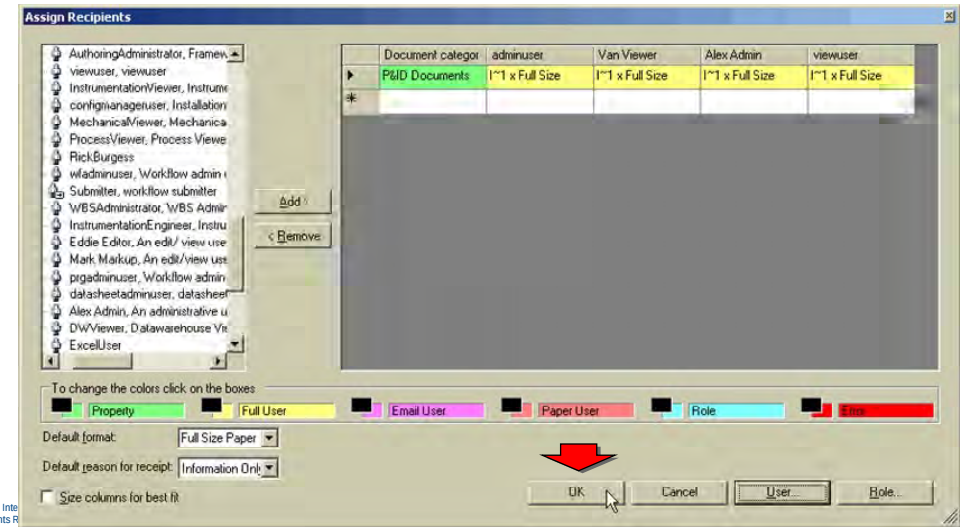


© 2005. Intergraph Corp.
All Rights Reserved.

To assign a recipient to the matrix, select the recipient in the recipient list and click **Add**.

Creating a Document Matrix

- **Click OK to complete the Document Matrix Assignments.**



© 2005. Inte
All Rights R

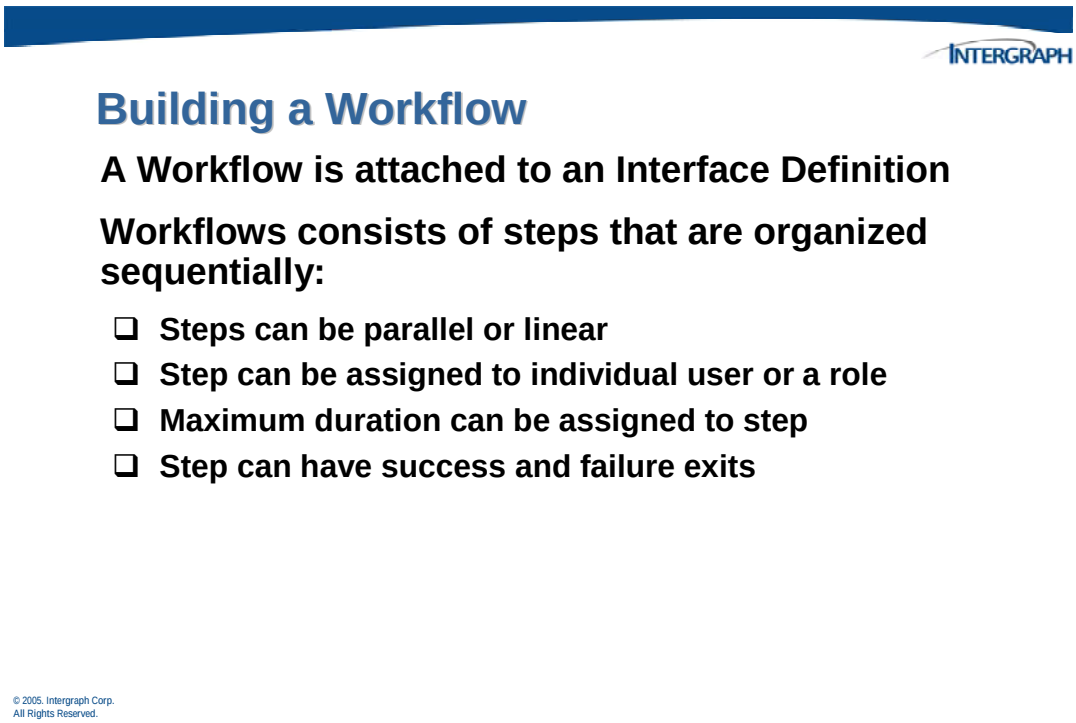
When the Internal Transmittal Matrix is complete, select the **OK** button on the *Assign Recipients* dialog.

Actions that can be performed on the Document Matrix:

- **Color palettes** - Allows you to customize the cell colors.
- **Users** - Allows you to search for users to assign to the distribution matrix.
- **Roles** - Allows you to search for roles to assign to the distribution matrix.
- **Default format** - Specifies the paper format automatically assigned to newly added paper users.
- **Default reason for receipt** - Specifies the reason for receipt automatically assigned to newly added users or users groups.
- **Size columns for best fit** - Resizes the columns in the matrix view.
- **Move Column** - Adjusts the order in which the user or role columns display.
- **Add Column** - Allows you to add a user or role column. You must type a valid user or group name to add the new column.
- **Duplicate Column** - Allows you to add a new user or role column with the same reasons for receipt as the column being duplicated.
- **Clear Column** - Removes all reasons for receipt from the column.
- **Delete Column** - Removes the user or role from the display.
- **Change Recipient** - Allows you to change the recipient.
- **Duplicate Row** - Adds a new row with the same properties and reasons for receipt as the row being duplicated.
- **Delete Row** - Removes the row from the display.

7.3 Building a Workflow

Workflows allow you to record a process using a template. Within a workflow, you can define the steps required to achieve the process and set planned and target dates for the step and the duration of the step.



Workflows contain the steps necessary to achieve a change management process. These steps can be serial, parallel, or a combination of both. Each step is assigned a sequence number, a duration period, and a user (Assignee) or user group. Each step can optionally have a list of checklist questions, which the user must perform at each step before completion of the change. The order of processing for each step is defined by the accept and reject steps. By default, the first step is started when the workflow is attached to the change object. Subsequent steps start when the prerequisite steps are completed.

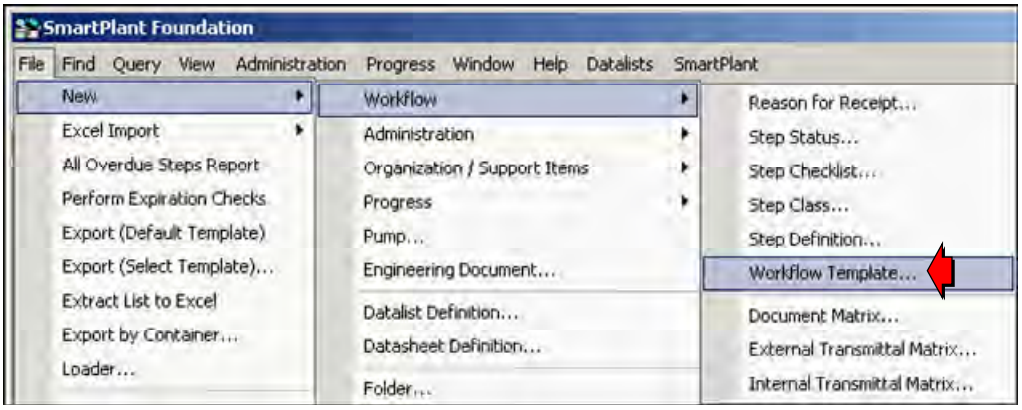
Parallel steps are configured by specifying a number of steps that start based upon the completion of a particular step.

The steps in the template workflow are selected from a library of step definitions to ensure a consistent look and feel to all workflows.

To build a new workflow:

Building a Workflow

- **From the menu, choose *File> New > Workflow > Workflow Template.***

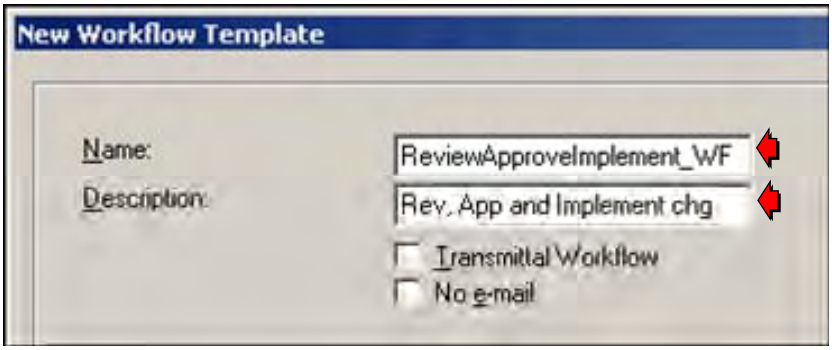


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Workflow Template* form will display.

Building a Workflow

- **Enter the *Name* and optional *Description* for the Workflow Template.**



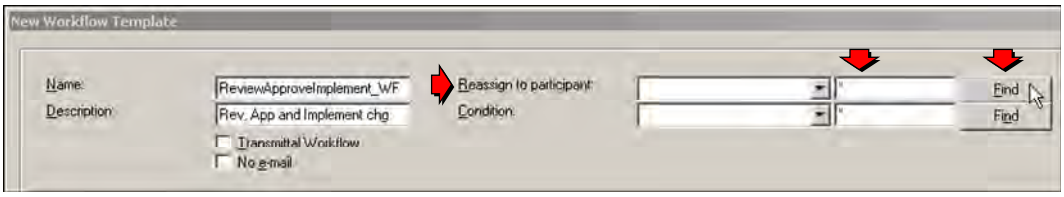
© 2005. Intergraph Corp.
All Rights Reserved.

The *New Workflow Template* form contains the following fields:

- **Name** - specifies the new workflow template name and is required
- **Description** - defines a short description for the workflow. This is an optional field.
- **Transmittal Workflow** - When this check box is selected, the *Transmittal Issue Step* box is displayed.
- **No E-mail** - enable the check box to disable all e-mail notifications to the recipients associated with this workflow

Building a Workflow

- **To display the *Reassign to Participant* list, type * or other suitable search criteria in the Reassign to Participant search box and click Find.**

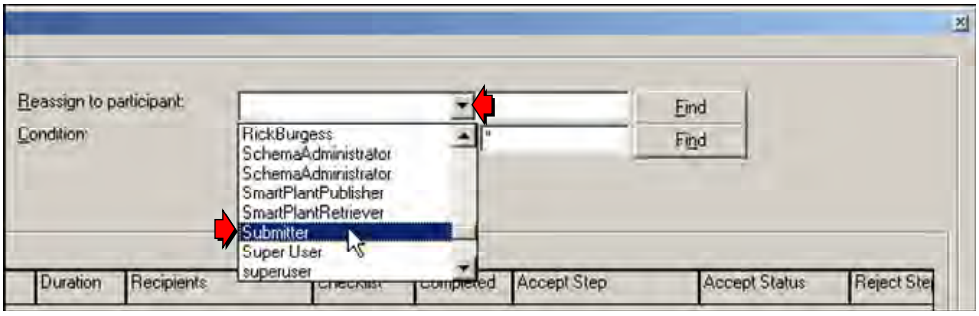


© 2005. Intergraph Corp.
All Rights Reserved.

To display the *Reassign to Participant* list, type * or other suitable search criteria in the search box and click **Find**.

Building a Workflow

- Click ▼ and select the participant from the list.

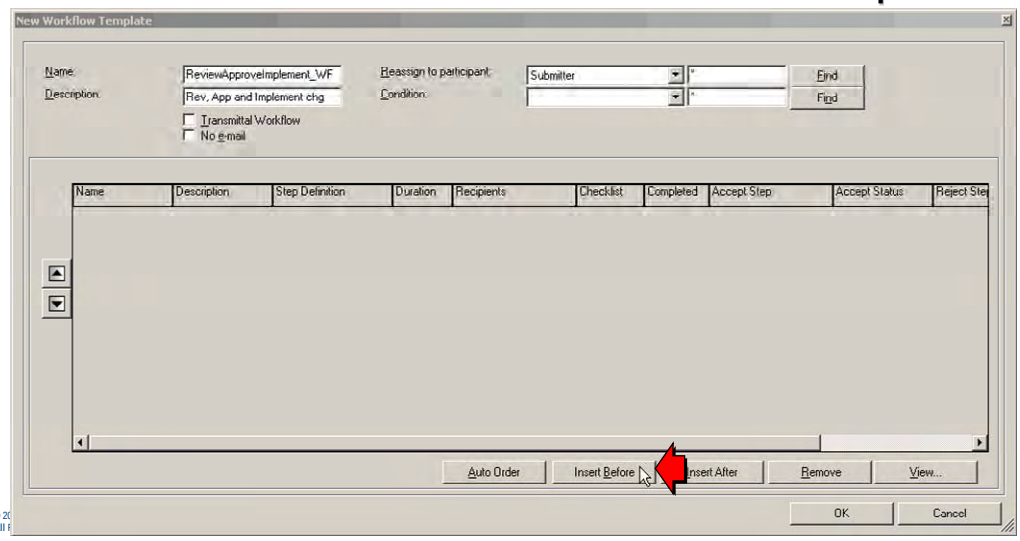


© 2005. Intergraph Corp.
All Rights Reserved.

The normal choice is **Submitter**, the person who submitted the object to the workflow.

Building a Workflow

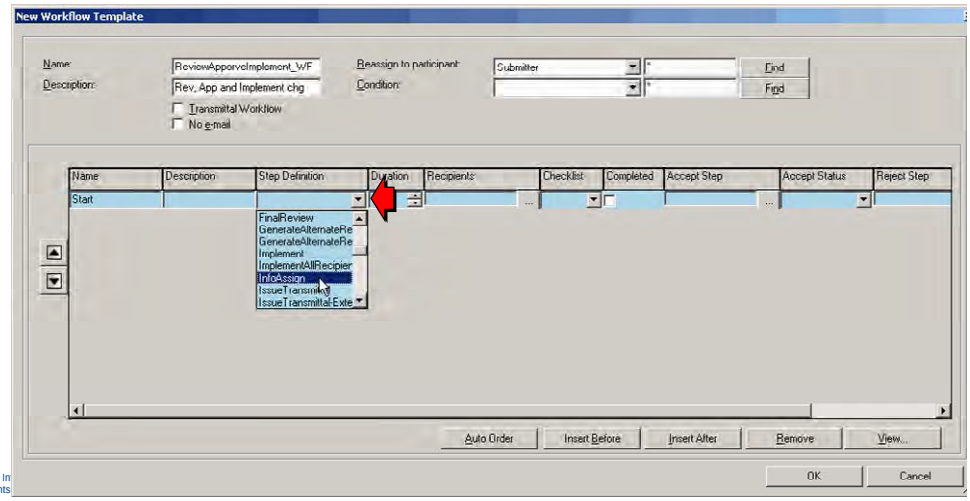
- Click Insert Before to create the first workflow step.



Choose a workflow step name to be used with this sequence number.

Building a Workflow

- **Select the Step Definition button and choose a workflow step from the displayed list.**



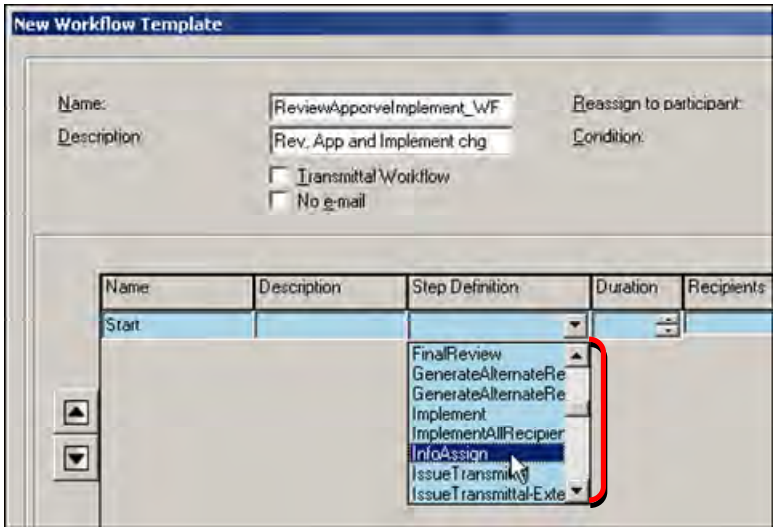
© 2005. In
All Rights

The *New Workflow Template* form fields (con't):

- **Step Definition** - specifies the step to be performed from the list. The *Name* box is populated automatically with the step name but can be edited.

Building a Workflow

Zoomed in view of the *Step Definition* selections.

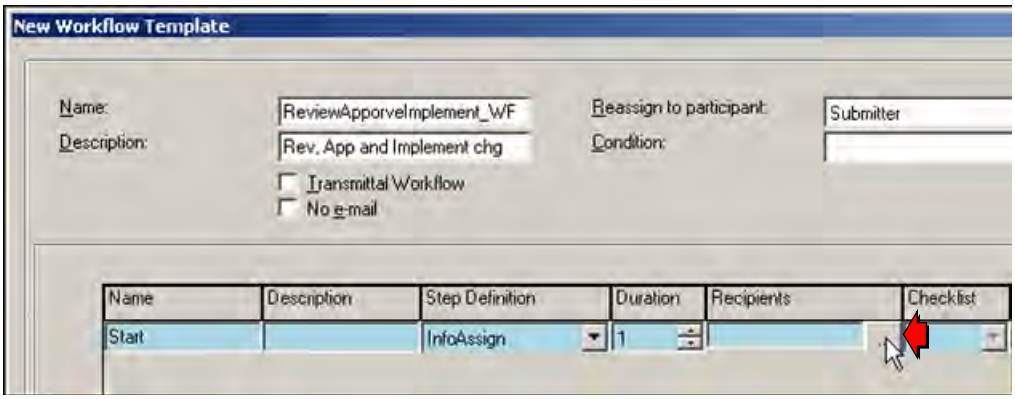


© 2005. Intergraph Corp.
All Rights Reserved.

Next to the **Recipients** box, click ... to display the *Update Step Recipients* form.

Building a Workflow

- **Choose the ellipses to select the recipient that will perform this workflow step.**



© 2005. Intergraph Corp.
All Rights Reserved.

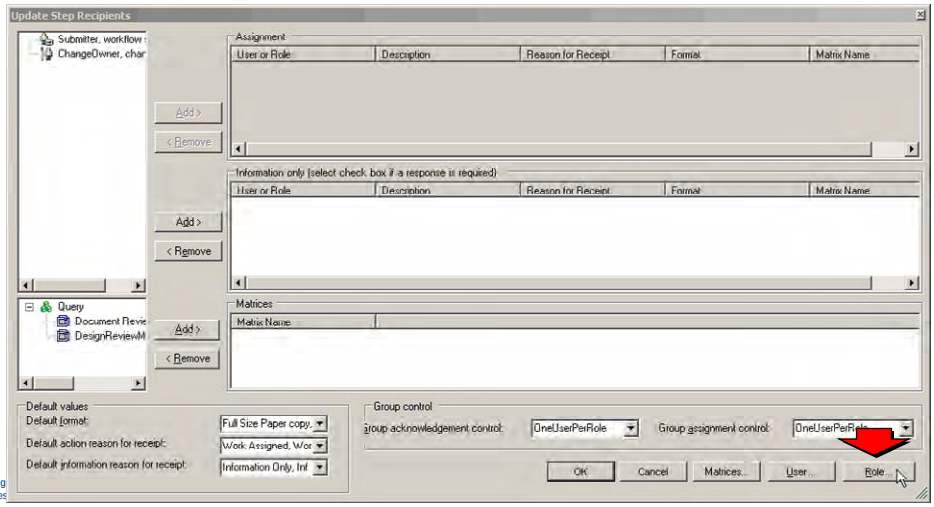
The *New Workflow Template* form fields (con't):

- **Duration** - specifies the duration range for the step. Scroll up or down numerically to select a duration range of working days. The duration range is from 1 to 60 working days and a value of zero is not allowed.
- **Recipients** – A dedicated GUI (*Update Step Recipients* form) is used to configure the recipients of a workflow step. Buttons are provided to query for *users*, *roles*, or *matrices*. The results of the search are added to the tree view on the left for drag and drop into the assignment or information set of recipients.

The *Update Step Recipients* form will display.

Building a Workflow

- **Click on the Roles button to display a list of available roles.**

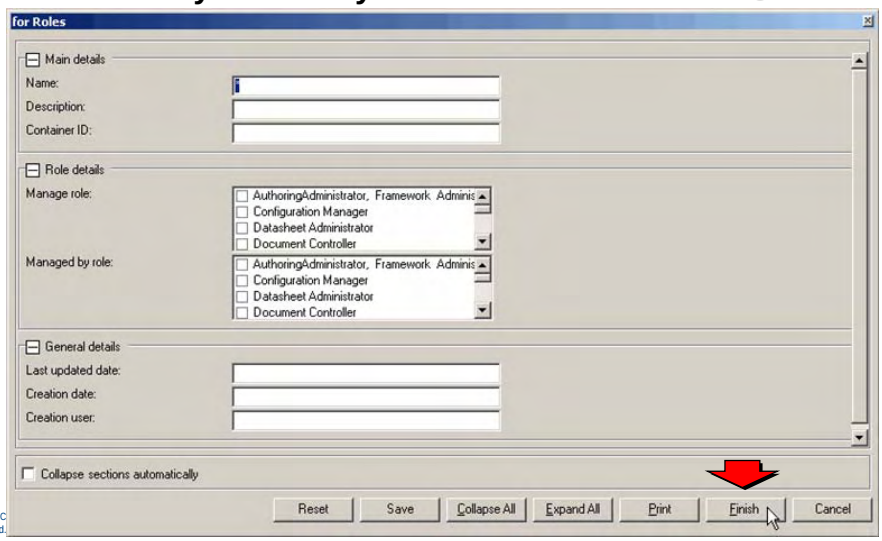


© 2005. Interg
All Rights Res

A *for Roles* dialog will display allowing a search for all or specific roles.

Building a Workflow

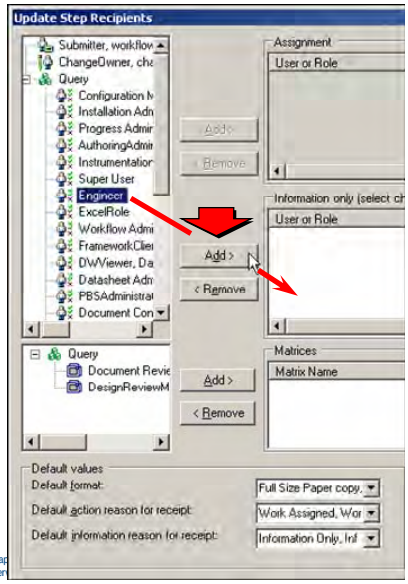
- **Enter any necessary search criteria and click OK.**



© 2005. Intergraph C
All Rights Reserved.

The *Tree View* will expand to show the results of the **Roles** query.

Building a Workflow



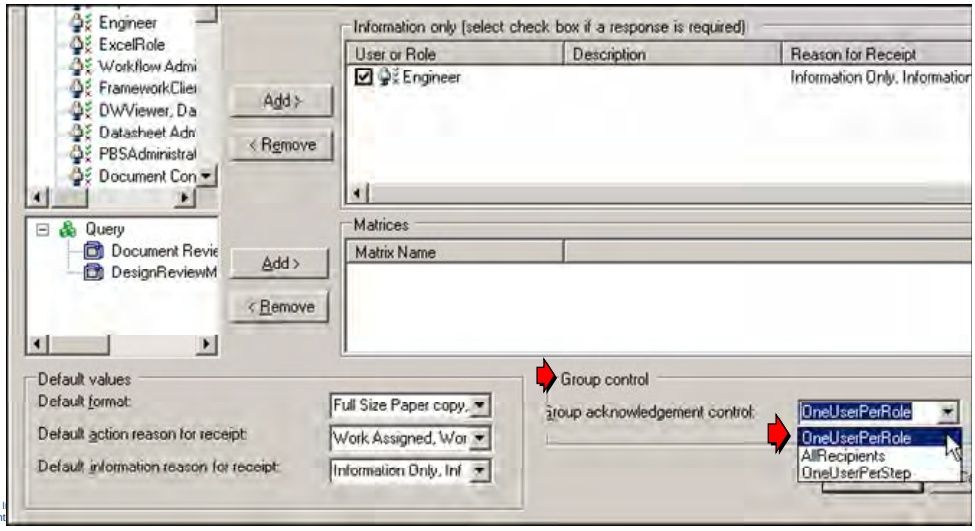
- **Choose a Role from the displayed list, then click Add.**

© 2005. Intergrap
All Rights Reser

You can set the *Group acknowledgment control* to **one** user or **all** recipients.

Building a Workflow

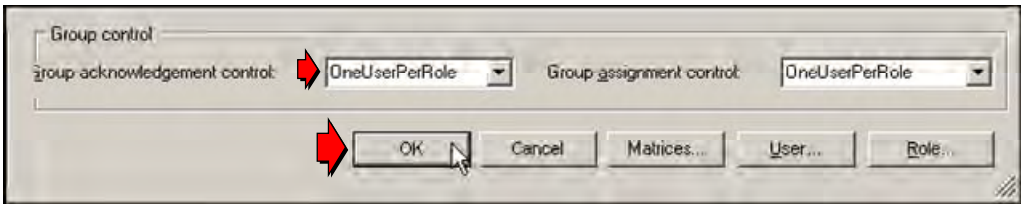
- **Select the *Group control* acknowledgement option.**



© 2005. I
All Right

Building a Workflow

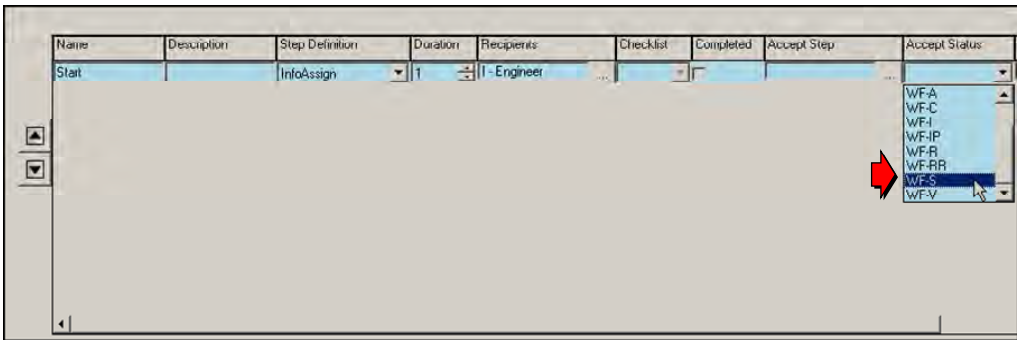
- Click **OK** to save the ***Step Recipients*** settings.



© 2005. Intergraph Corp.
All Rights Reserved.

Building a Workflow

- Select the **Accept Status** button to choose an existing workflow status.



© 2005. Intergraph Corp.
All Rights Reserved.

The *New Workflow Template* form fields (con't):

- **Status** - specifies a workflow Status from the list. The default statuses are:
  - **!WF-C** – Workflows that are InProgress

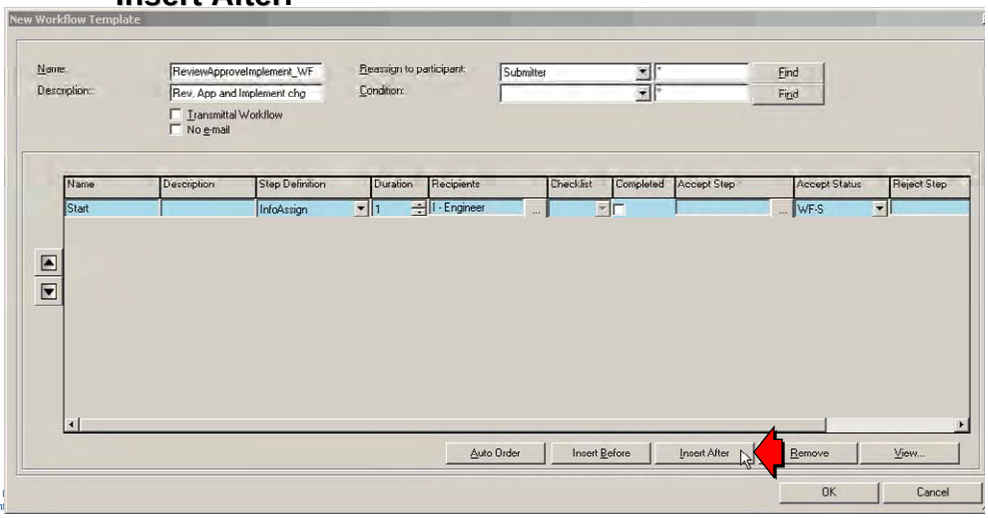
- **WF** – In workflow
- **WF-A** – Approved in workflow
- **WF-C** – Workflow completed
- **WF-I** – Issued in workflow
- **WF-R** – Rejected in workflow
- **WF-V** – Validated in workflow

And the new ones that were added:

- **WF-IP** – Object InProgress in Workflow
- **WF-RR** – Ready for Review Rejected
- **WF-S** – Workflow Started

Building a Workflow

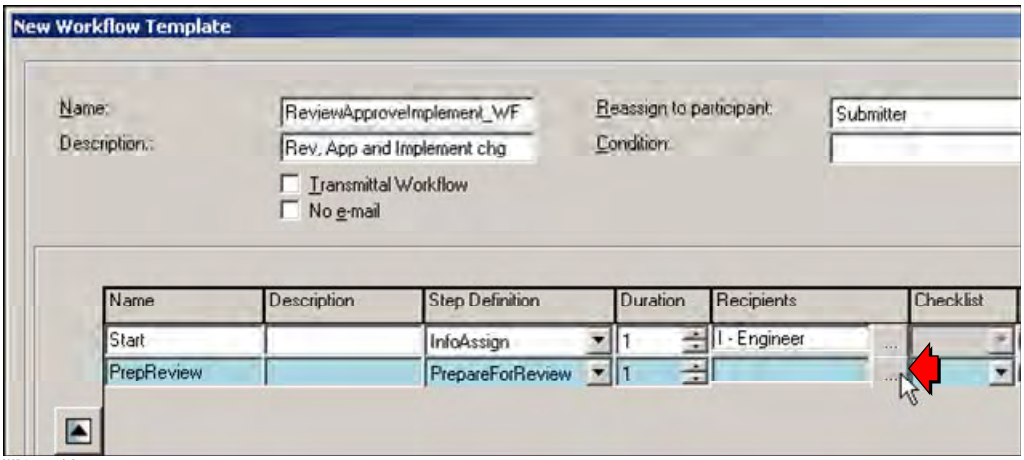
- **To add the next step, select the current step and click Insert After.**



Continue defining the next step in the workflow by clicking **Insert After** to add a new step after the current selection.

Building a Workflow

- **Choose the ellipses to select the recipient that will perform this workflow step.**

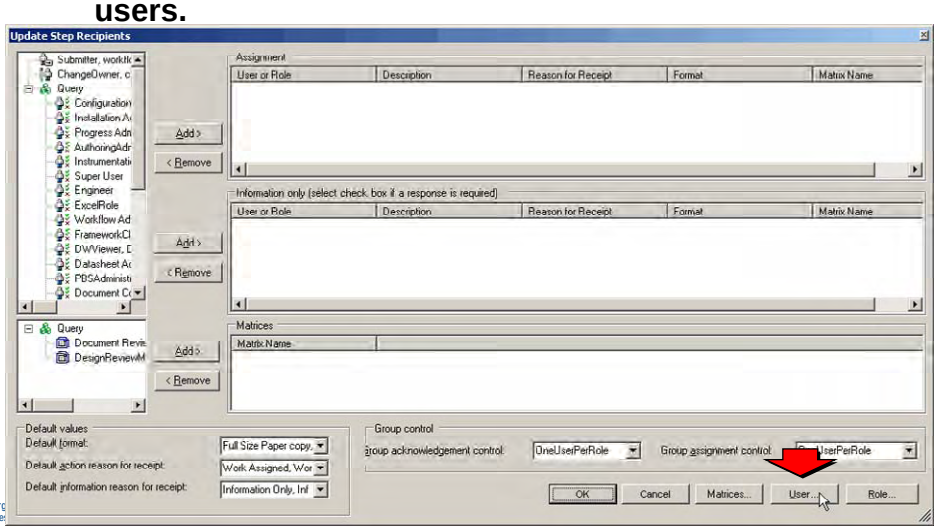


© 2005. Intergraph Corp.
All Rights Reserved.

The *Update Step Recipients* form will display.

Building a Workflow

- **Click on the Users button to display a list of available users.**



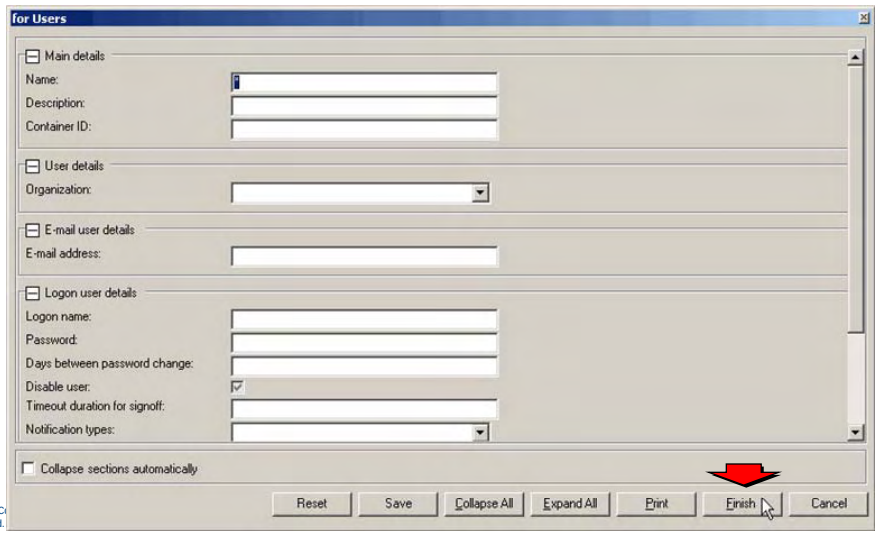
© 2005. Interg
All Rights Re

Users can be also be added as recipients.

The *Query for Users* form will appear.

Building a Workflow

- **Enter any necessary search criteria and click OK.**

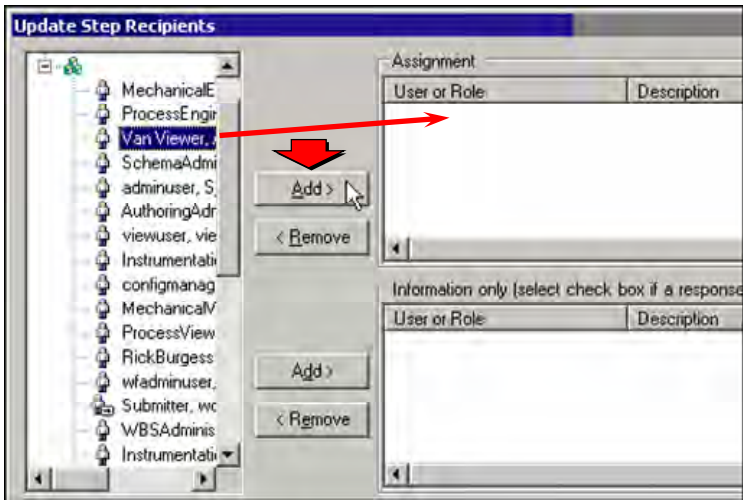


© 2005. Intergraph Corp.
All Rights Reserved.

The *Query for Users* area of the tree is populated with the results of the query.

Building a Workflow

- **Choose a *recipient* User from the displayed list, then click Add.**

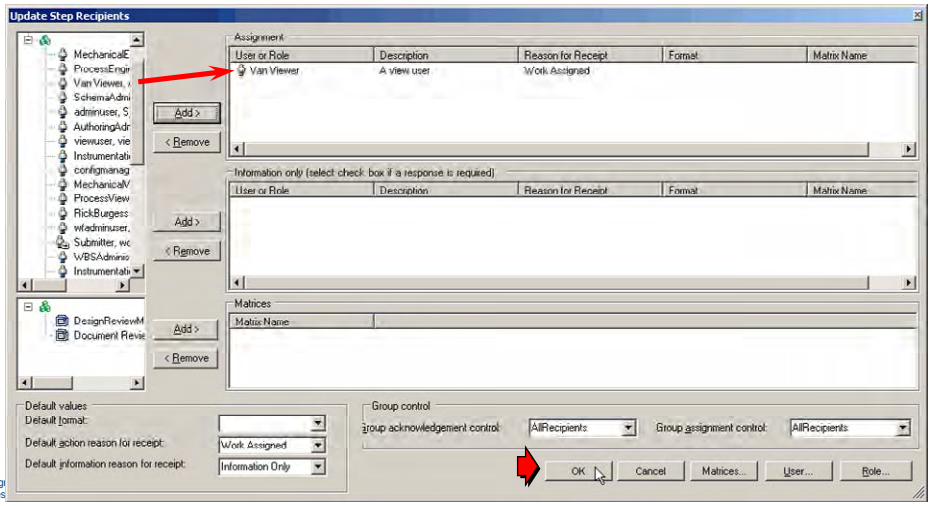


© 2005. Intergraph Corp.
All Rights Reserved.

In this example, *Van Viewer* is being added as a recipient to a different step.

Building a Workflow

- **The selected user is displayed in the *Assignment* field, then click OK.**

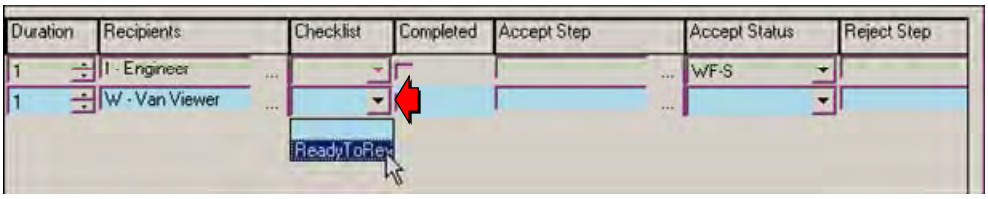


© 2005. Intergr
All Rights Res

This step will have a checklist associated with it.

Building a Workflow

- **Select the Check Lists button to choose an existing check list to associate with this step.**



© 2005. Intergraph Corp.
All Rights Reserved.

The *New Workflow Template* form fields (con't):

- **Checklists** – assigns an existing checklist to the step. Select a checklist from the list in the **Checklists** field.

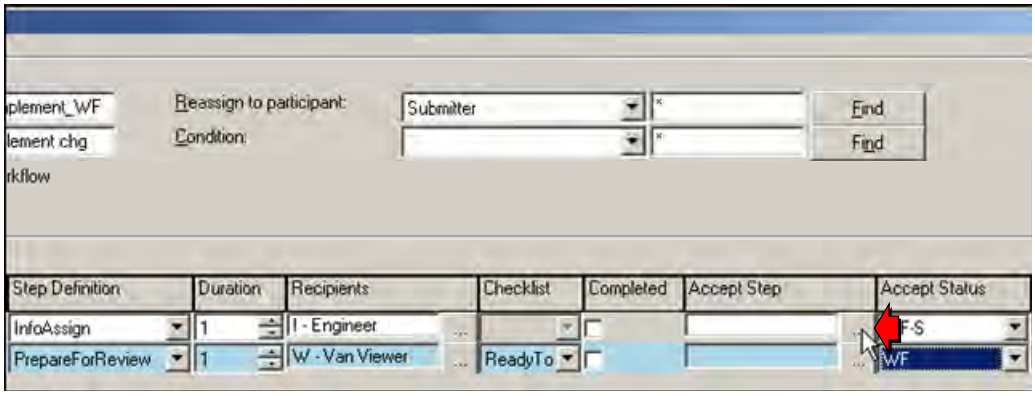
- **Completed** - specifies that the requirements for this step must be completed. This indicates to the user, via the *To Do List*, that all checklist items must be completed by the user if a checklist is attached to this step. If a checklist is specified for the step, but if the Complete button is not checked, then the user can complete some, none, or all of the check list items.

To set the **Accept Status**, the status of the workflow object if the step is completed successfully, click ▼ and select the status from the list.

INTERGRAPH

Building a Workflow

- **Choose the ellipses to select the Accept Step that will be associated with this workflow step.**

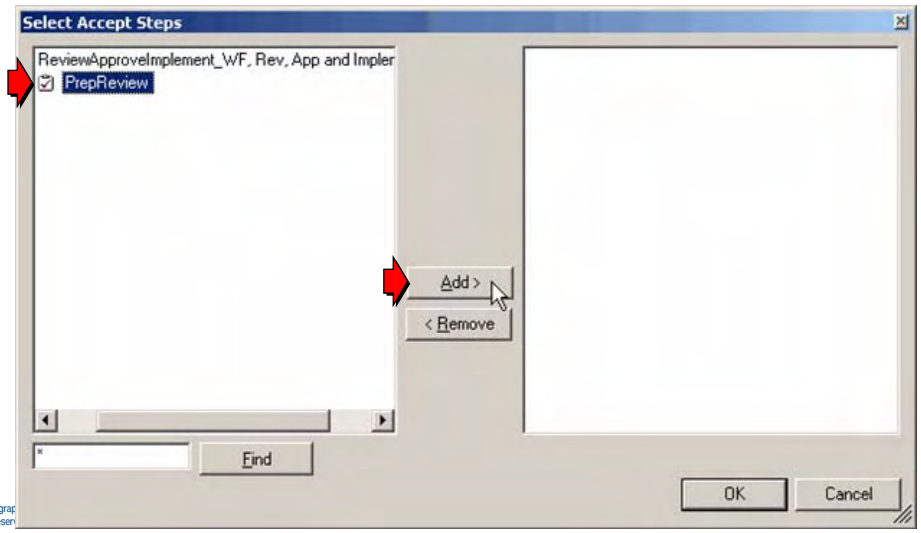


© 2005. Intergraph Corp.
All Rights Reserved.

When the **Accept Step** ellipsis is selected, a *Select Accept Steps* dialog is displayed.

Building a Workflow

- **Select the accept step (Workflow step) and click Add.**

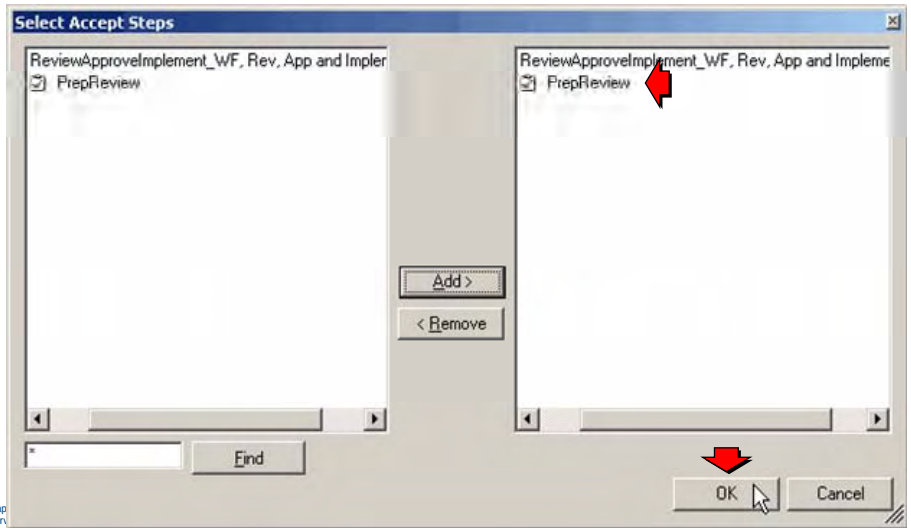


© 2005. Intergrap
All Rights Reserv

The added step will display in the right pane.

Building a Workflow

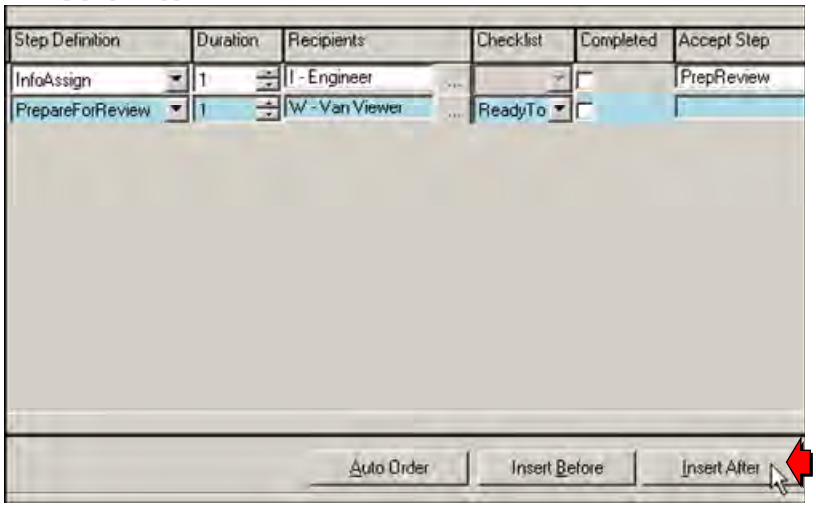
- **Once the selected step displays in the right pane, click OK.**



© 2005. Intergrap
All Rights Reserv

Building a Workflow

- **To add the next step, select the current step and click Insert After.**

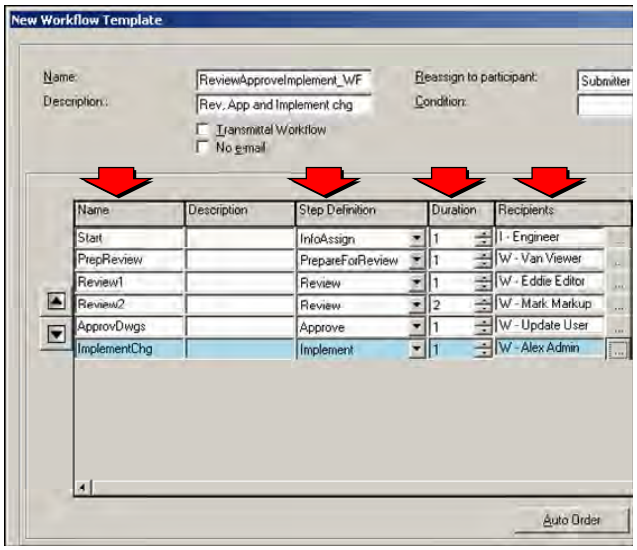


© 2005. Intergraph Corp.
All Rights Reserved.

Continue to build the workflow template by adding the next steps.

Building a Workflow

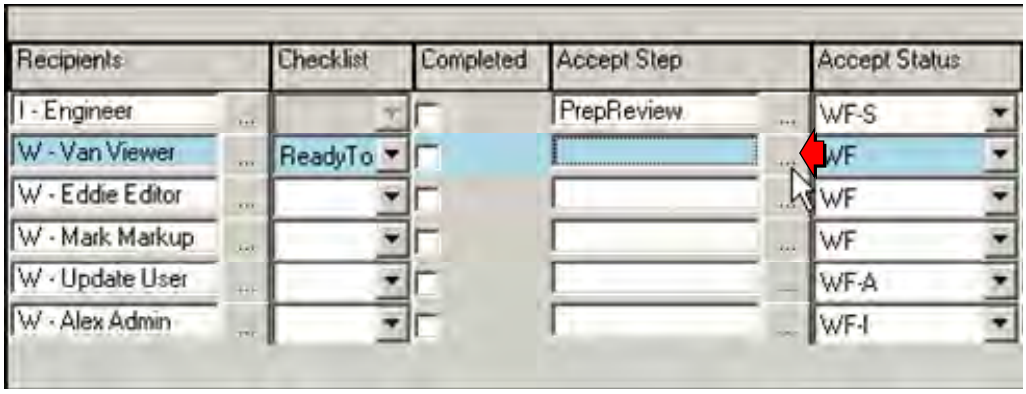
- **Continue defining the workflow steps and recipients.**



© 2005. Intergraph Corp.
All Rights Reserved.

Building a Workflow

- **Choose the ellipses to select the Accept Step that will be associated with this workflow step.**

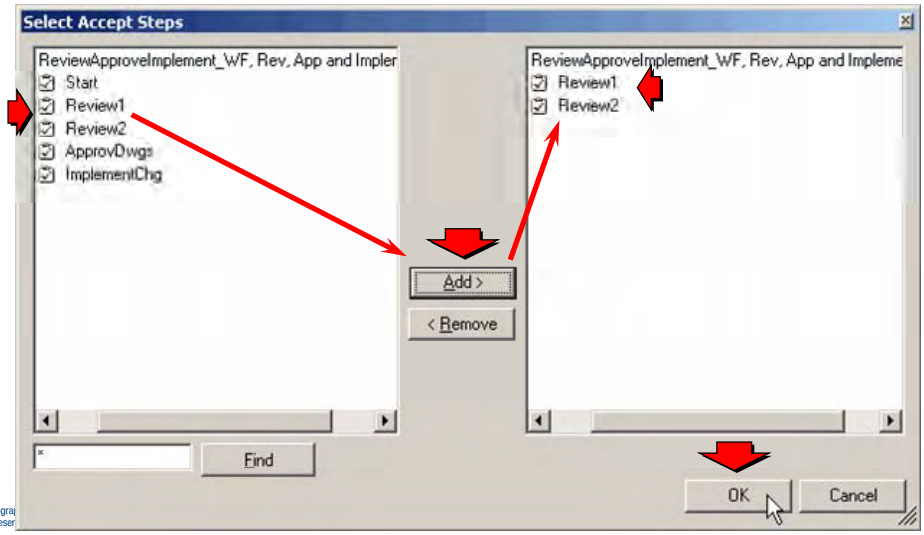


© 2005. Intergraph Corp.
All Rights Reserved.

Parallel steps can be chosen as a dependency for a particular step, such as Review**1** and **Review2** being dependencies for **ApprovDwgs**.

Building a Workflow

- **Select the accept step(s) (one at a time) and click Add.**

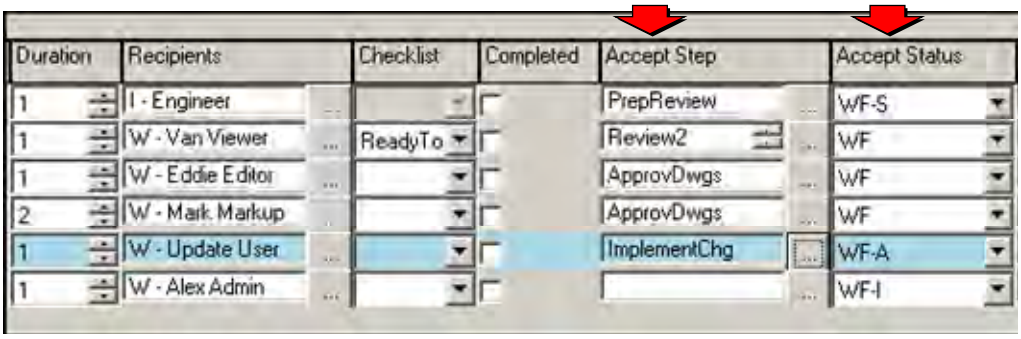


© 2005. Intergra
All Rights Reser

The *Accept Step* field now shows both **Review1 & Review 2**, which means that **ApprovDwgs** cannot be performed until Review1 and Review2 are complete.

Building a Workflow

- **Continue to set the *Accept Steps* and *Accept Statuses*.**

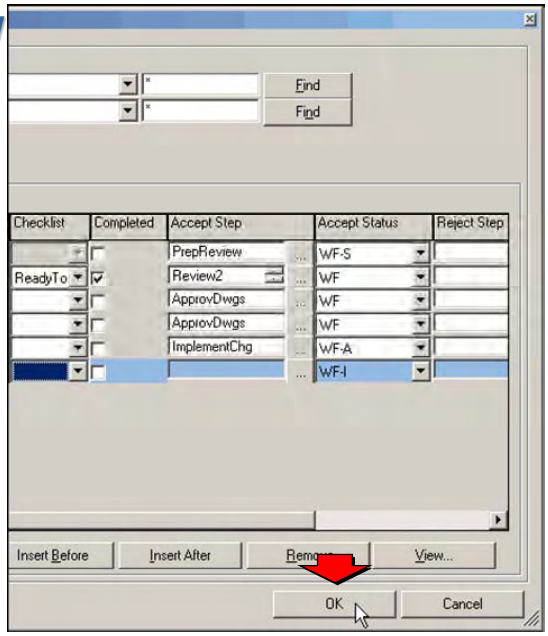


© 2005. Intergraph Corp.
All Rights Reserved.

Continue to define the structure for the workflow template.

Building a Workflow

- **When all of the steps have been defined for the workflow template, click OK.**



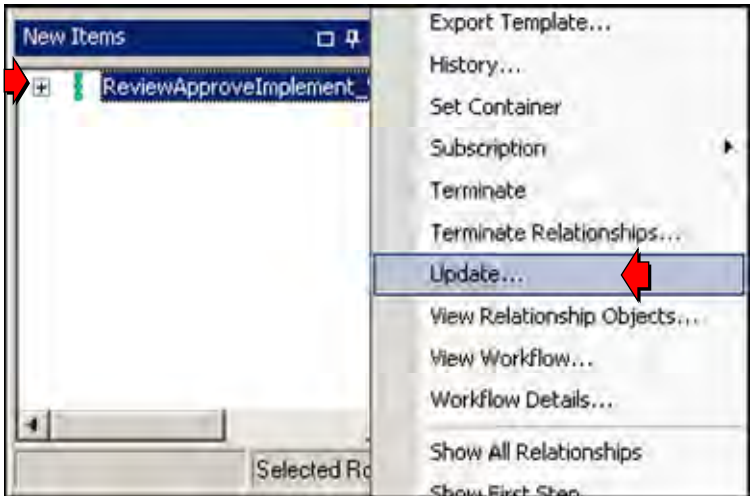
© 2005. Intergraph Corp.
All Rights Reserved.

7.4 Viewing Workflows

Once the workflow has been created, a graphical display can be used to review the workflow steps.

Building a Workflow

- **Right-click the ReviewApproveImplement_WF Workflow and click *Update* from the dynamic menu.**

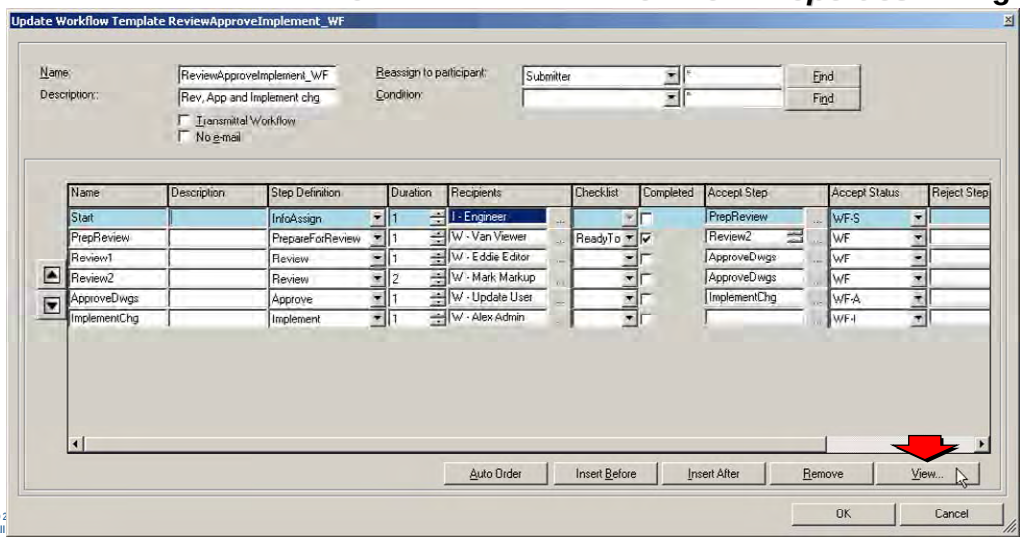


© 2005. Intergraph Corp.
All Rights Reserved.

The *Update Workflow Template* form will display.

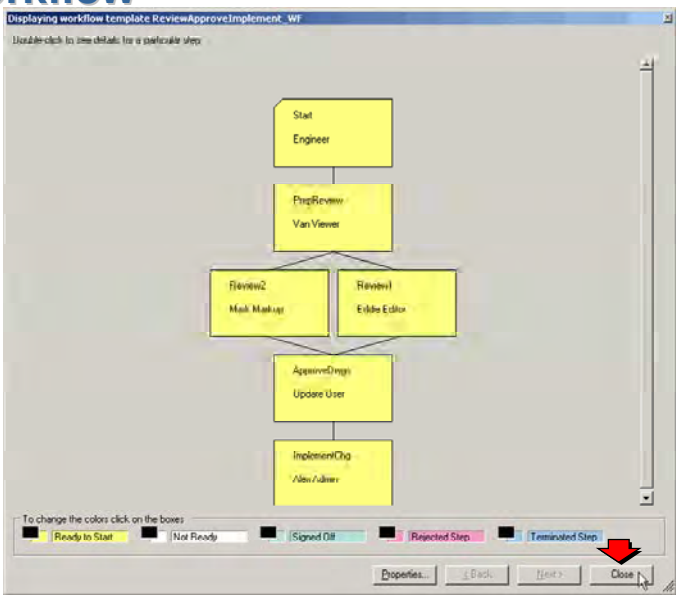
Building a Workflow

- **Choose the View button on the *Workflow Properties* dialog**



Building a Workflow

- **Click Close to close the graphical workflow view.**

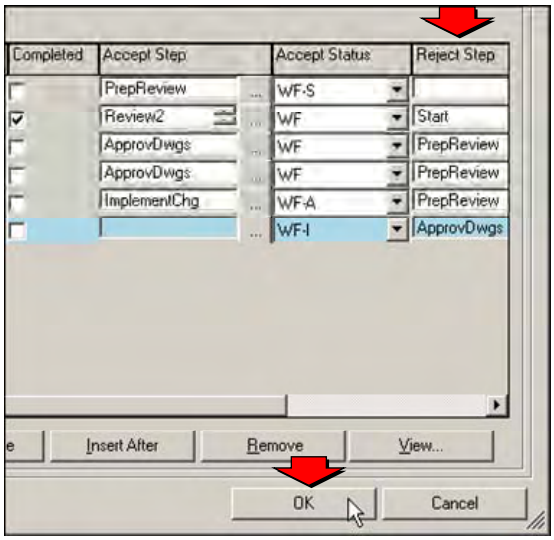


© 2005. Intergraph Corp.
All Rights Reserved.

Each node in the graphical display represents a step in the workflow and the status of the step is represented by a color. In the example above, the color **white** represents a state of **Not Ready**.

Building a Workflow

- **Set the *Reject Steps* and *Reject Statuses* then click OK.**



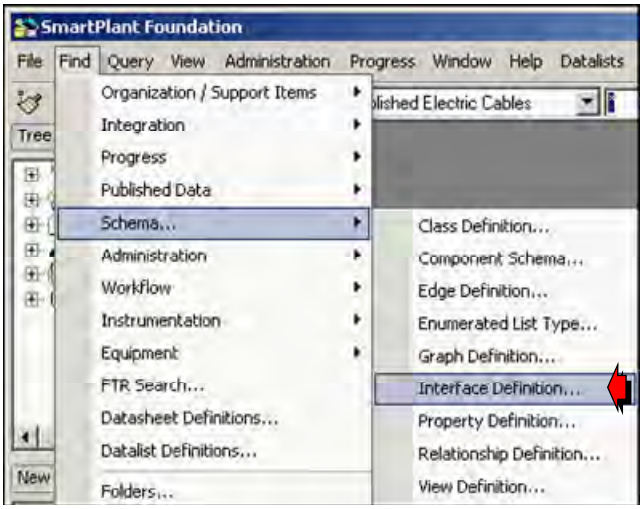
© 2005. Intergraph Corp.
All Rights Reserved.

7.5 Attaching Workflows

Attaching a workflow to an interface definition means that the workflow is available for all objects or classes associated with the interface definition. For example, if you attach a workflow to the **ISPFDocumentRevision** interface definition, all classes that realize the **ISPFDocumentRevision** interface definition can be submitted to the selected workflow.

Attaching a Workflow

- **From the menu, click the *Find > Schema > Interface Definition* command.**



© 2005. Intergraph Corp.
All Rights Reserved.

A search will be performed to locate the *ISPFDocumentRevision* interface. This is because when a document object is created, in reality three object instances are created – the document master, the document revision and the document version. The ISPFDocumentRevision interface is associated with the revision object that is created. When an object gets submitted to a workflow, it's not the document master that is routed through the workflow steps, it's a specific **version** of the document revision.

The *Find Interface Definition* dialog box displays.

Attaching a Workflow

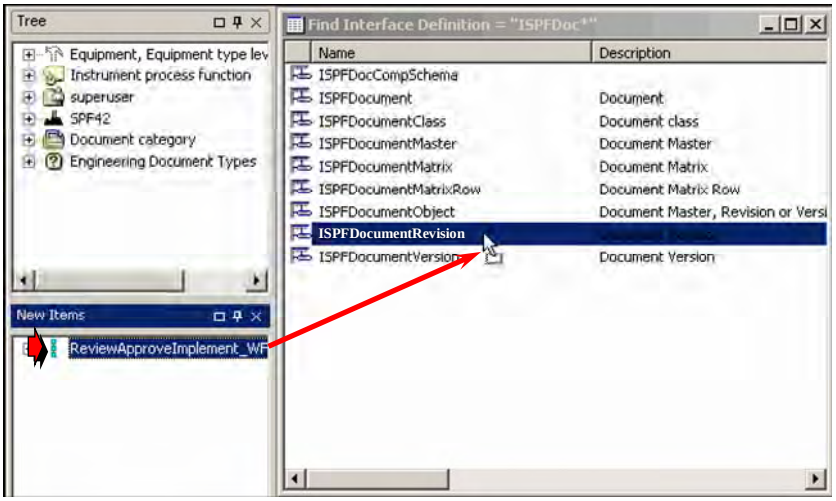
- **Type ISPFDoc* in the *Find Interface Definition* dialog box and click OK.**

© 2005. Intergraph Corp.
All Rights Reserved.

Select the **ReviewApproveImplement_WF** workflow in the *New Items* window.

Attaching a Workflow

- **Drag and drop a custom workflow onto the *ISPFDocumentRevision* interface.**

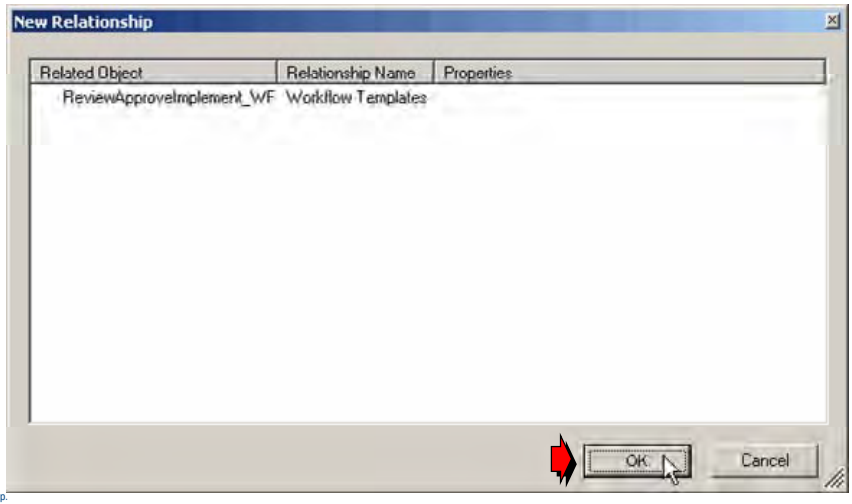


© 2005. Intergraph Corp.
All Rights Reserved.

Select the custom workflow, and drag it onto the *ISPFDocumentRevision* InterfaceDef in the browser window.

Attaching a Workflow

- **Verify that the new relationship is correct and click OK.**

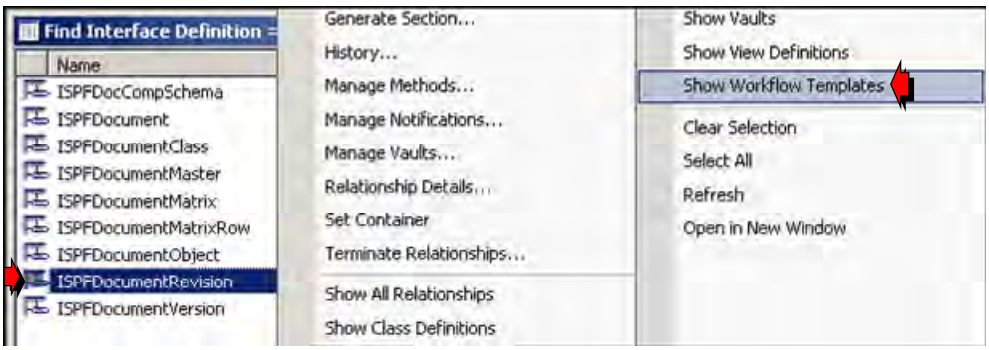


© 2005. Intergraph Corp.
All Rights Reserved.

Next, view the workflows that are related to the interface ISPFDocumentRevision.

Attaching a Workflow

- **Right-click the ISPFDocumentRevision InterfaceDef and click *Show Workflow Templates* on the shortcut menu.**

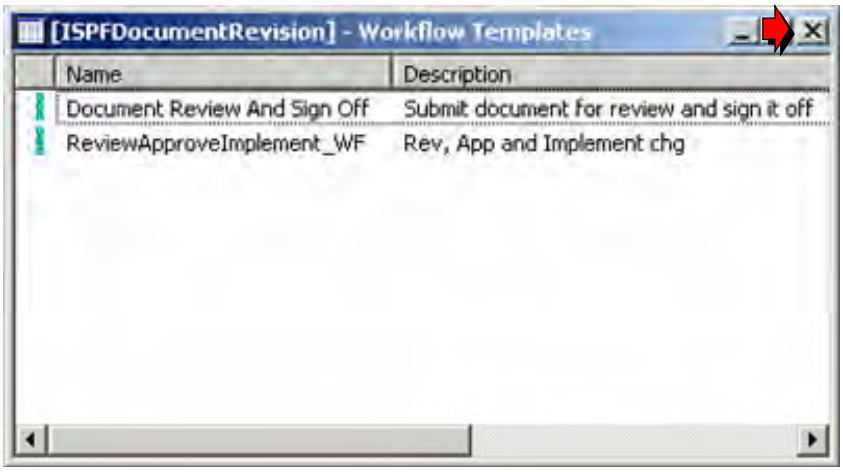


© 2005. Intergraph Corp.
All Rights Reserved.

Repeat this procedure for any additional workflows that you want to be available to select from the Desktop Client.

Attaching a Workflow

- **Click on the X to close the Workflow Templates window.**



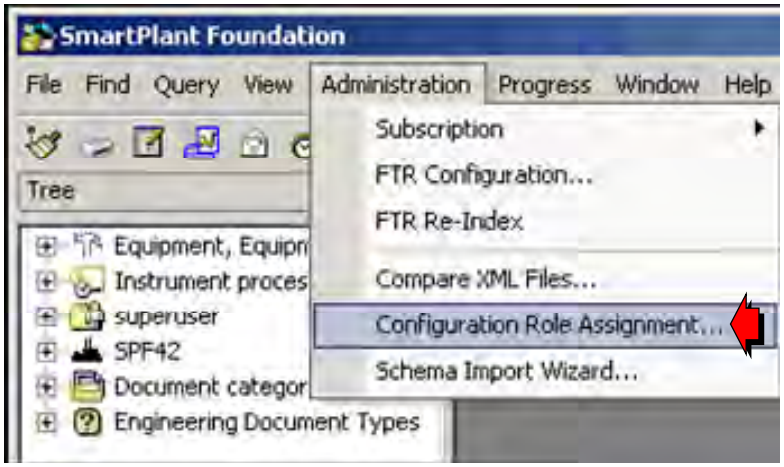
© 2005. Intergraph Corp.
All Rights Reserved.

7.5.1 Workflow Access Configuration

Access control of functionality and objects within SmartPlant Foundation is controlled by the relationship of *Users* to *Roles* and *Role Assignments*.

Workflow Access Configuration

- **Select *Administration > Configuration Role Assignment* from the menu.**

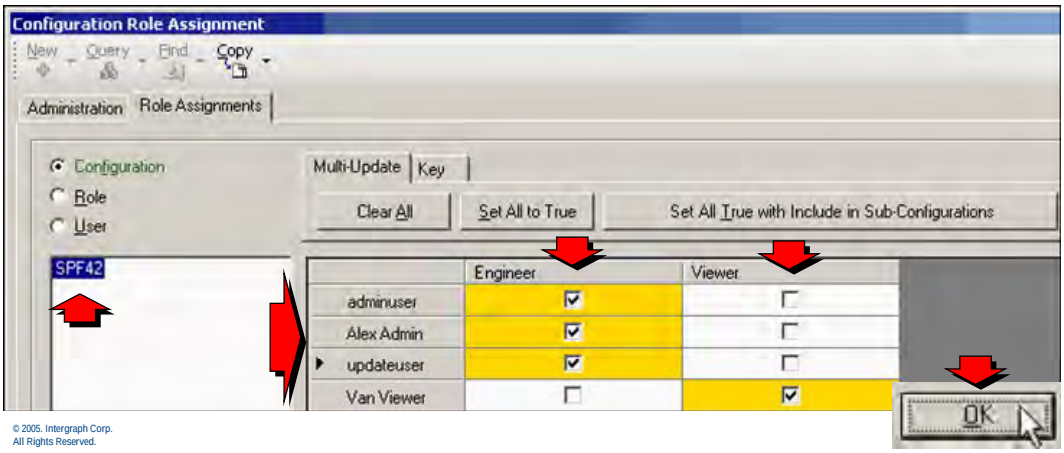


© 2005. Intergraph Corp.
All Rights Reserved.

In order for users to participate in the workflow process, they must be assigned a role that has access to the necessary commands.

Workflow Access Configuration

- **Select the type of view to use for making role assignments and then enable the toggle in the matrix to assign a specific user to a specific role within this configuration.**

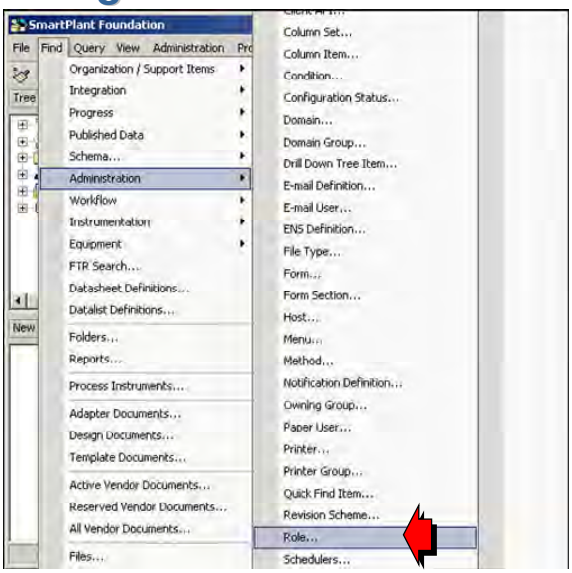


© 2005. Intergraph Corp.
All Rights Reserved.

Additional Access Groups may need to be added to certain roles. In the following example, the *WorkflowUpdate* access group will be added to the *Viewer* role.

Workflow Access Configuration

- **From the menu, click *Find > Administration > Role.***

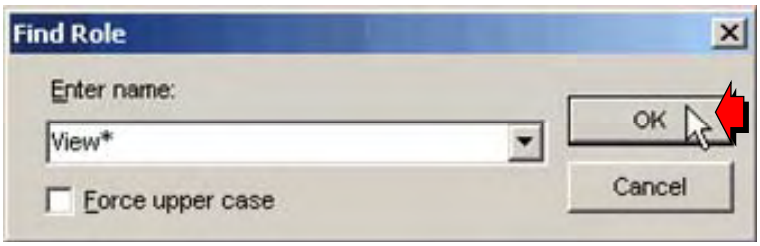


© 2005. Intergraph Corp.
All Rights Reserved.

The *Find Role* dialog will appear.

Workflow Access Configuration

- **Select OK to find and display the *Viewer* role.**

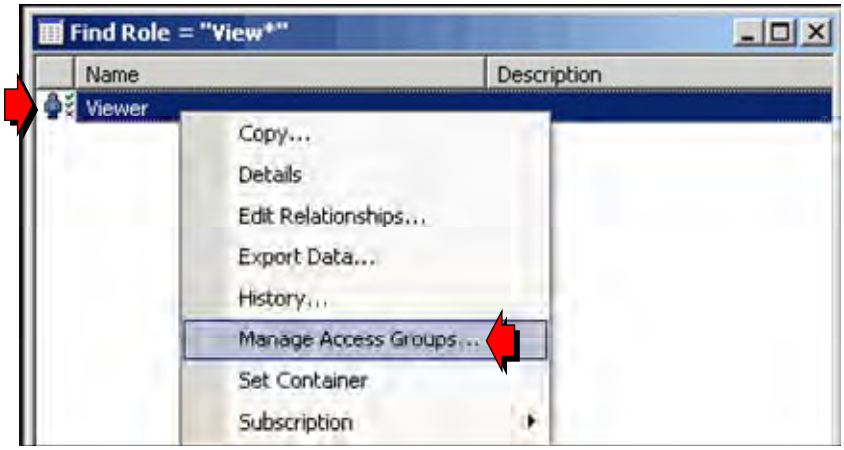


© 2005. Intergraph Corp.
All Rights Reserved.

To relate Access Groups, use the ***Manage Access Groups*** shortcut command from the role.

Workflow Access Configuration

- **Right click on the role and from the shortcut menu, select *Manage Access Groups.***

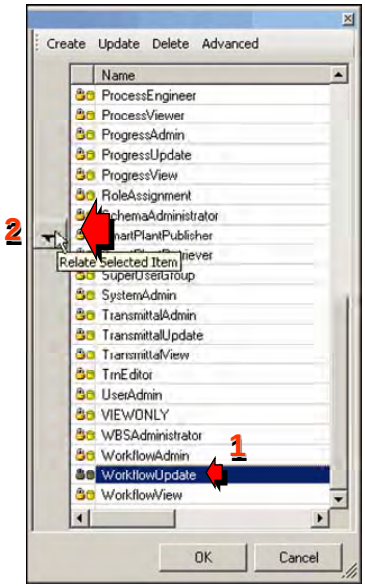


© 2005. Intergraph Corp.
All Rights Reserved.

The *Manage Access Groups* form will display.

Workflow Access Configuration

- **On the right side of the dialog, select the *WorkflowUpdate* access group to be related to the role and click the left arrow button.**

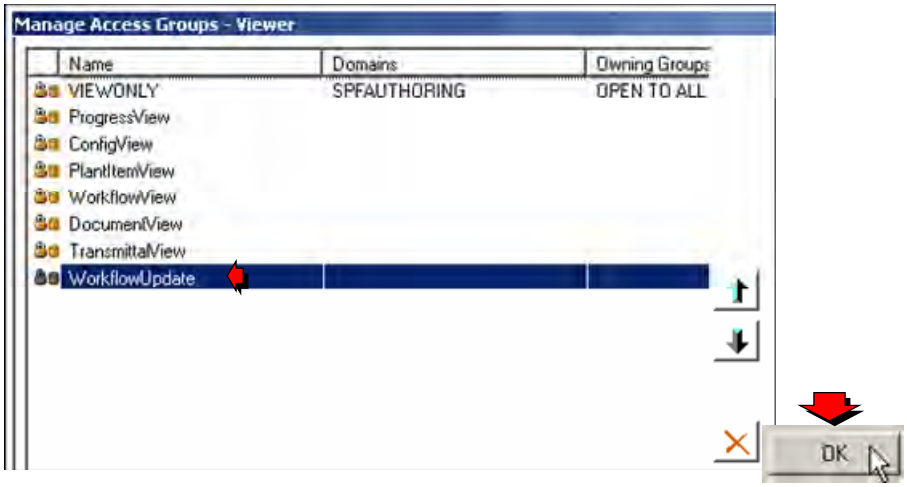


© 2005. Intergraph Corp.
All Rights Reserved.

The selected access groups will be "dragged" to the left side of the form as shown below.

Workflow Access Configuration

- **Select OK once all the relationships have been selected and configured.**

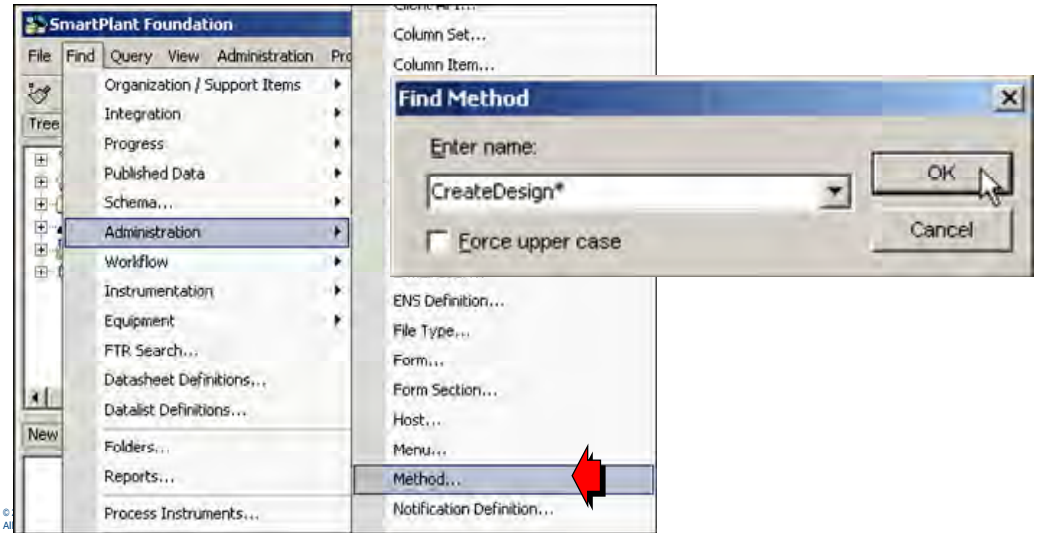


© 2005. Intergraph Corp.
All Rights Reserved.

In order to attach a workflow as a new document is created, the *CreateDesignDocument* method will need to be modified.

Workflow Access Configuration

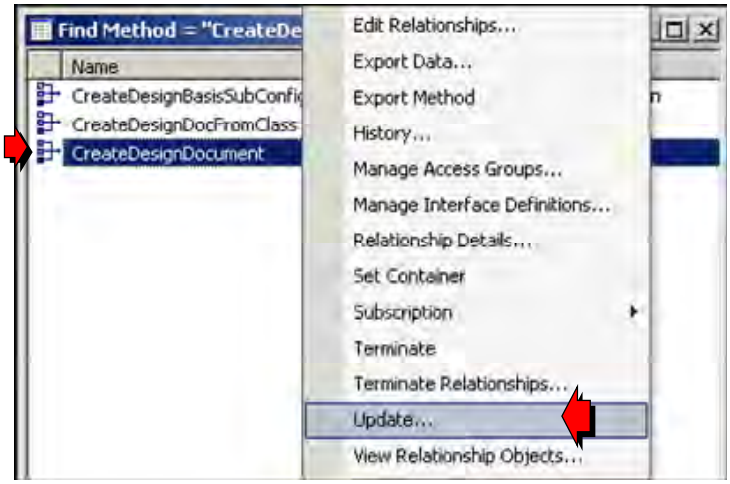
- **From the menu, click *Find > Administration > Method***



Once the method gas been displayed in a list view window, use the update command to make the modification.

Workflow Access Configuration

- **Right click on the method and from the shortcut menu, select *Update.***

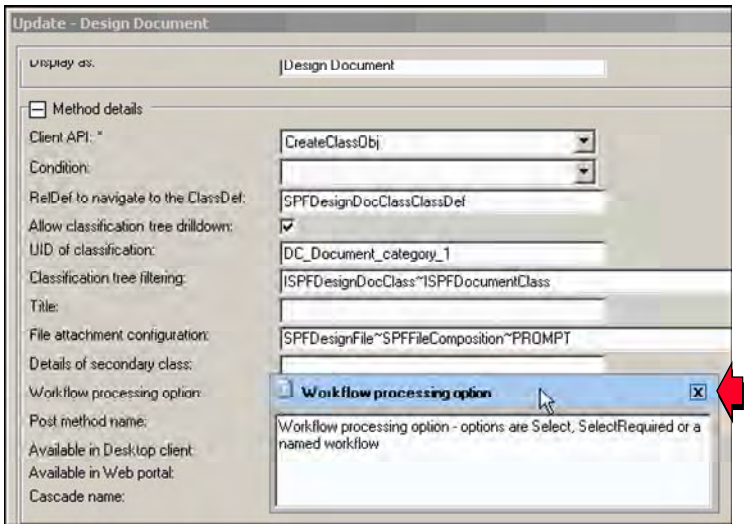


© 2005. Intergraph Corp.
All Rights Reserved.

Hover the cursor over the *Workflow processing option* field to display the help window.

Workflow Access Configuration

- **Click the "*X*" to dismiss the help window.**

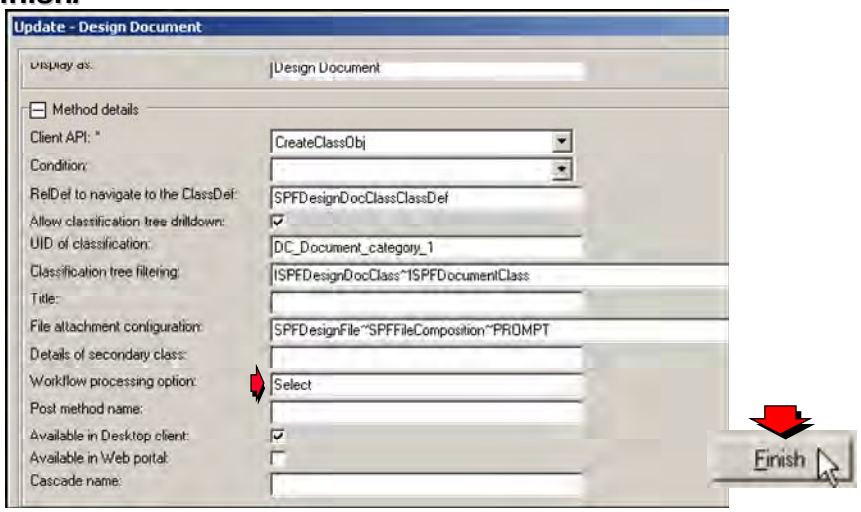


© 2005. Intergraph Corp.
All Rights Reserved.

Edit the method to make the necessary changes to enable workflow processing.

Workflow Access Configuration

- **Enter *Select* for the *Workflow processing option* and click Finish.**



© 2005. Intergraph Corp.
All Rights Reserved.

7.6 Workflow Options

Workflow options are implemented via the **Find** administration sub menu options. They provide configurable e-mail notifications that are distributed.

Workflow options provide a mechanism to configure the default e-mail message sent to the recipients of a step. These are overridden by any e-mails linked to the step class.

When a new step is started in a workflow, the workflow engine determines if the step class (relating to the current step) has an associated e-mail message and uses that e-mail message to send out a workflow e-mail notification. If no e-mail message is found, then the default e-mail message is generated from the *Options* command.

INTERGRAPH

Workflow Options

The "Options" function allows you to configure emails, statuses, access control and process control.

You can configure the following options:

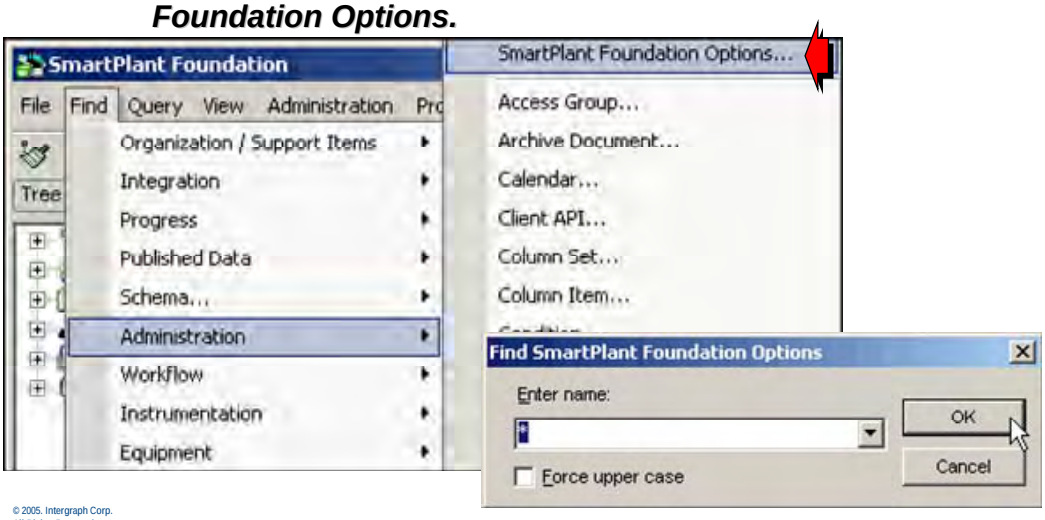
- **E-mail Options**
- **Overdue Options**

© 2005. Intergraph Corp.
All Rights Reserved.

Perform a search to locate and display the SmartPlant Foundation Options object.

Workflow Options

- **From the menu, click *Find > Administration > SmartPlant Foundation Options.***

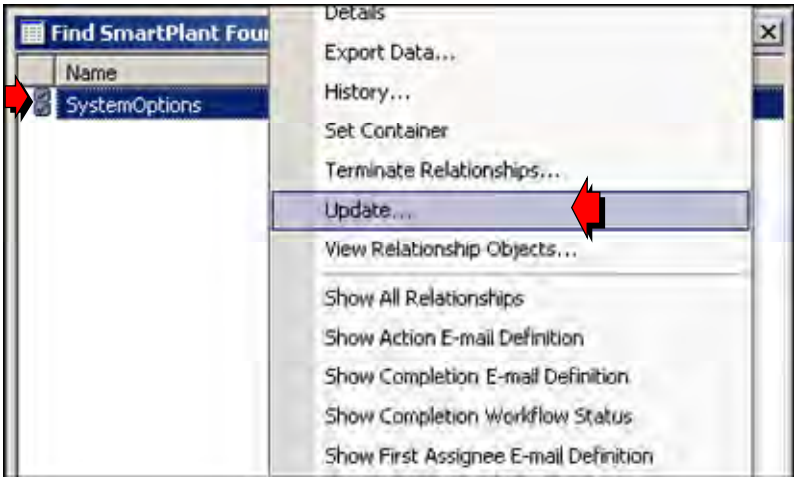


© 2005. Intergraph Corp.
All Rights Reserved.

Use the ***Update*** command to make any changes or additions to the SPF Options object.

Workflow Options

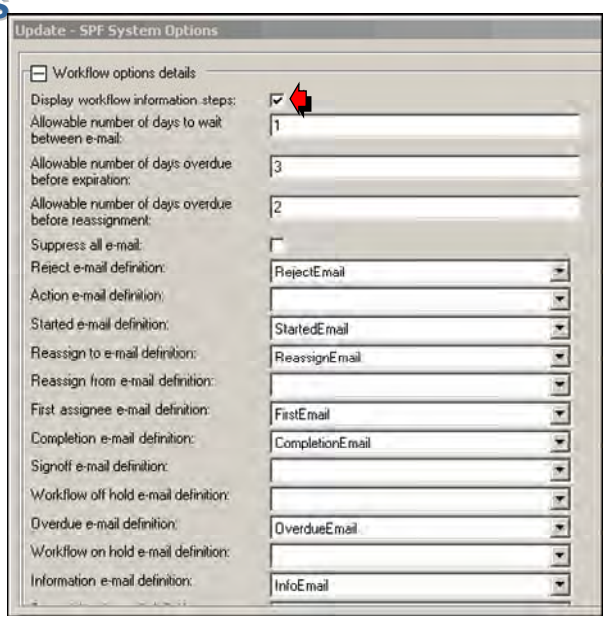
- **Right click on the method and from the shortcut menu, select *Update.***



© 2005. Intergraph Corp.
All Rights Reserved.

Workflow Options

- **Configure any additional options for use in SPF Workflow.**



© 2005. Intergraph Corp.
All Rights Reserved.

Below is a description of the definitions for the SPF Options object:

- In the **Allowable number of days to wait between e-mail** box, type the number of days to wait between sending overdue reminder e- mails.

  For example, if the interval is two days, the system will only send an e-mail reminder every two days instead of every day.

- In the **Allowable number of days overdue before expiration box**, type the number of days a step can exceed its target date before the process server begins the expiration process.

- In the **Allowable number of days overdue before reassignment** box, type the number of days a step can be overdue before it is reassigned.

- **Suppress all e-mail** - Option to suppress all e-mail notifications from workflow to recipients.

- **Reject e-mail definition** - The e-mail message sent to recipients when a step is rejected.

- **Action e-mail definition** - The e-mail message sent to the recipients when a step action is required.

- **Started e-mail definition** - The default e-mail message sent to the recipients of a step when that step is started. The delivery of this e-mail is optional.

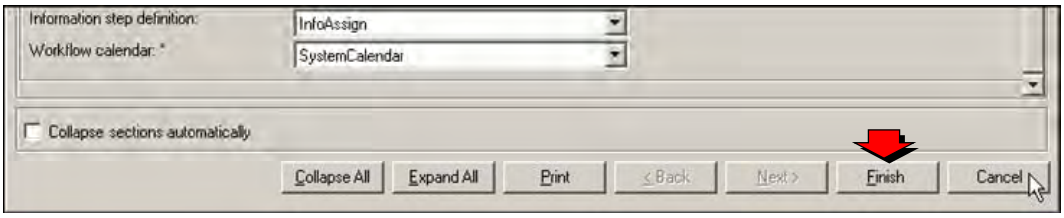
- **Reassign to e-mail definition** - The e-mail message sent to the recipient to which the step is reassigned.

- **Reassign from e-mail definition** - The e-mail message sent to the recipient the step is reassigned from.
- **First assignee e-mail definition** - The e-mail message sent to the first assignee.
- **Completion e-mail definition** - The e-mail message sent to the workflow submitter when the workflow is completed. The delivery of this e-mail is optional.
- **Sign off e-mail definition** - The e-mail message sent to recipients when a step is signed off.
- **Workflow off hold e-mail definition** - The e-mail message sent to recipients when the workflow is taken off hold.
- **Overdue e-mail definition** - The e-mail message sent to recipients when a step is overdue.
- **Workflow on hold e-mail definition** - The e-mail message sent to recipients when the workflow is put on hold.
- **Information e-mail definition** - The e-mail message sent to recipients when an information step is assigned to them.
- **Step claimed e-mail definition** - The e-mail message sent to the recipients when a step is claimed.
- **Future actions e-mail definition** - Option to configure an e-mail message that is sent to all recipients of all future steps in a workflow when the first step has been completed. This provides the recipients with a notification of potential future work.

INTERGRAPH

Workflow Options

- **Select Finish once all the workflow options have been selected and configured.**



© 2005. Intergraph Corp.
All Rights Reserved.

7.7 Using Workflows

Workflows allow you to record a process using a template. A user can add a workflow to an interface definition for which no workflows have been previously configured. When submitting an object to a workflow, they will then be presented with a list of all workflows available in the system for the class of the selected item.

Since the workflow has been configured to send email notifications, open the email client.

Using a Workflow

- **Double-click on the Outlook Express email icon from the Windows desktop.**

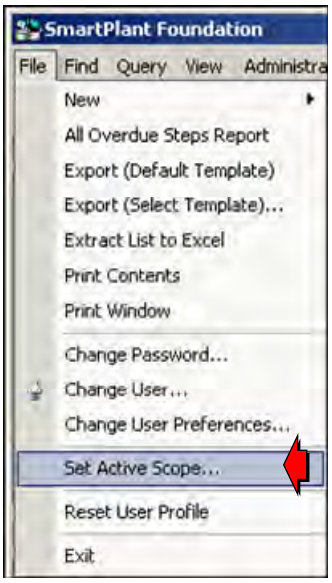
© 2005. Intergraph Corp.
All Rights Reserved.

7.7.1 Setting the Active Scope

Before creating objects or searching for objects, a user must first set the active scope. This will specify a *Plant*, a *Role* and a *Create Scope* with which the user will be confined for the operations.

In order to set the correct scope, use the ***File*** > ***Change User*** command to log in as user **alex**.

Using a Workflow



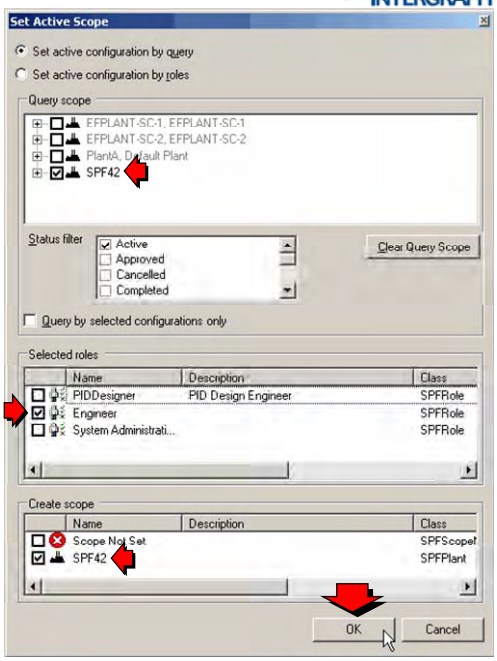
© 2005. Intergraph Corp.
All Rights Reserved.

- **Select the *Set Active Scope* command from the desktop client *File* menu.**

The *Set Active Scope* dialog will display.

Using a Workflow

- **Configure the appropriate *Plant*, *Role* and *Create scope*, then click OK.**



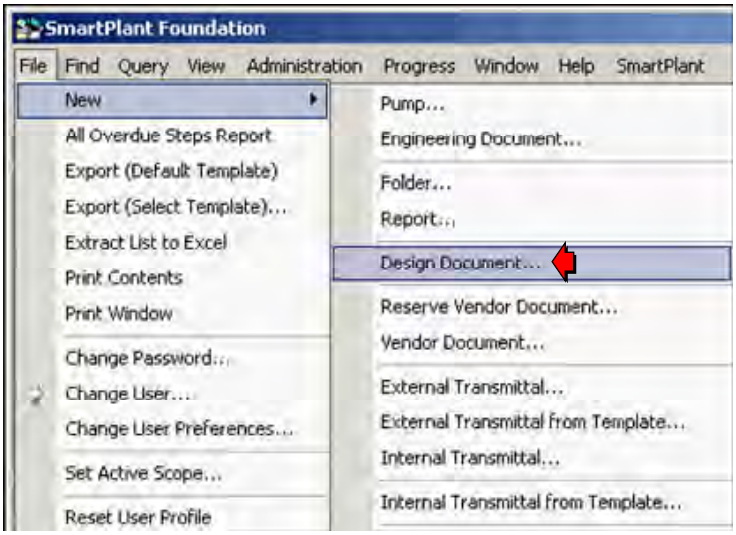
© 2005. Intergraph Corp.
All Rights Reserved.

7.8 Submitting a Document to the Workflow

The first step is to create an object, such as a document, which will be used by the workflow in a custom review/approval process.

Using a Workflow

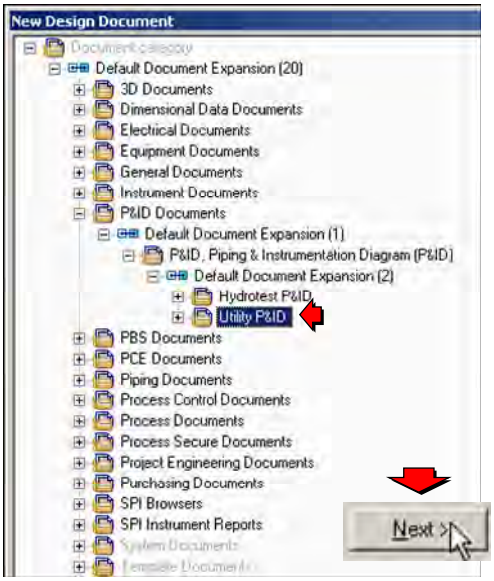
- **Select *File > New> Design Document* from the menu.**



© 2005. Intergraph Corp.
All Rights Reserved.

A *New Design Document* window will display.

Using a Workflow

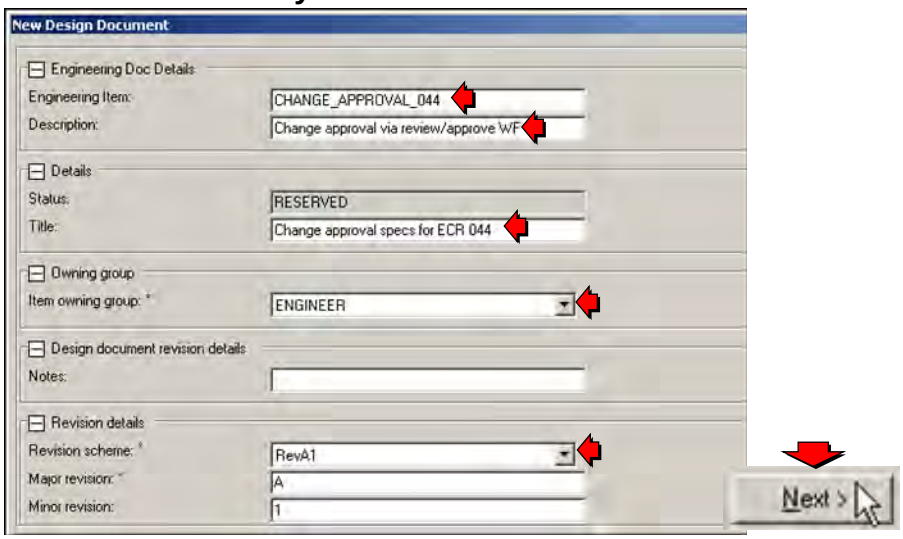


- **Expand the *Document category* tree then highlight Utility P&ID and click Next.**

The *New Design Document* create document form will display.

Using a Workflow

- **Enter the necessary new document values and click Next.**

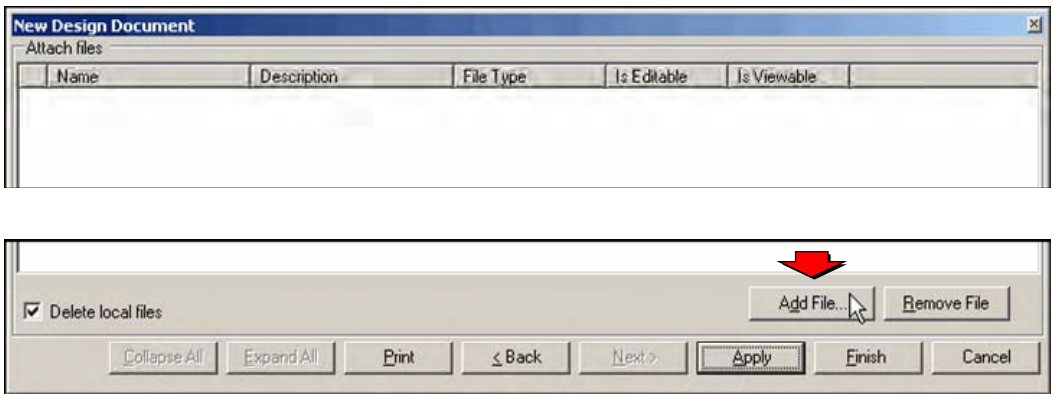


© 2005. Intergraph Corp.
All Rights Reserved.

An electronic file can be attached to the document and processed by the workflow. The *Attach files* dialog will display allowing you to attach a file to this new document. This procedure can also be performed at a later time.

Using a Workflow

- **Select the *Add File* button to browse for a file.**

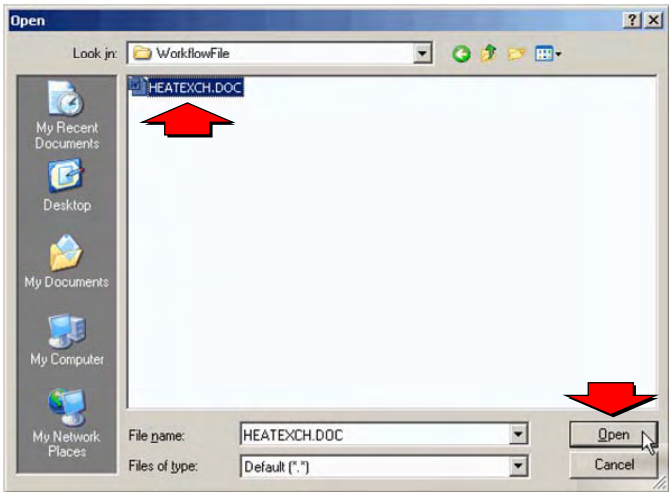


© 2005. Intergraph Corp.
All Rights Reserved.

A folder browser window will display. Use this browser to locate the file that will be attached to the new document. The file can reside in any folder on the client.

Using a Workflow

- **Open a folder and choose a file to be attached then select *Open*.**



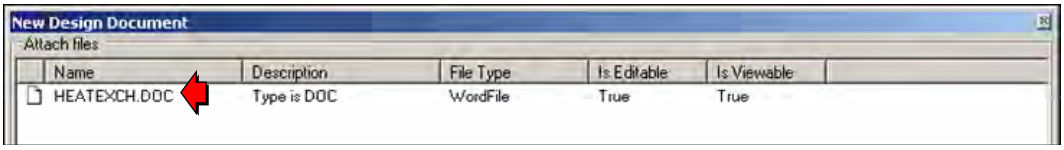
© 2005. Intergraph Corp.
All Rights Reserved.

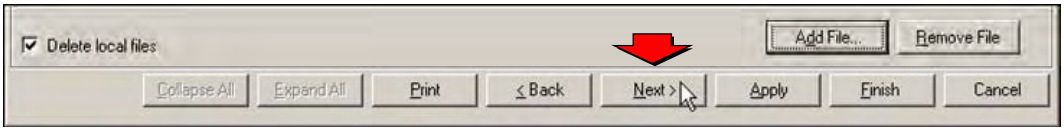
The selected file will be displayed in the *Attach files* window.

INTERGRAPH

Using a Workflow

- **Verify the correct file is listed in the *Attach files* dialog and click Next.**

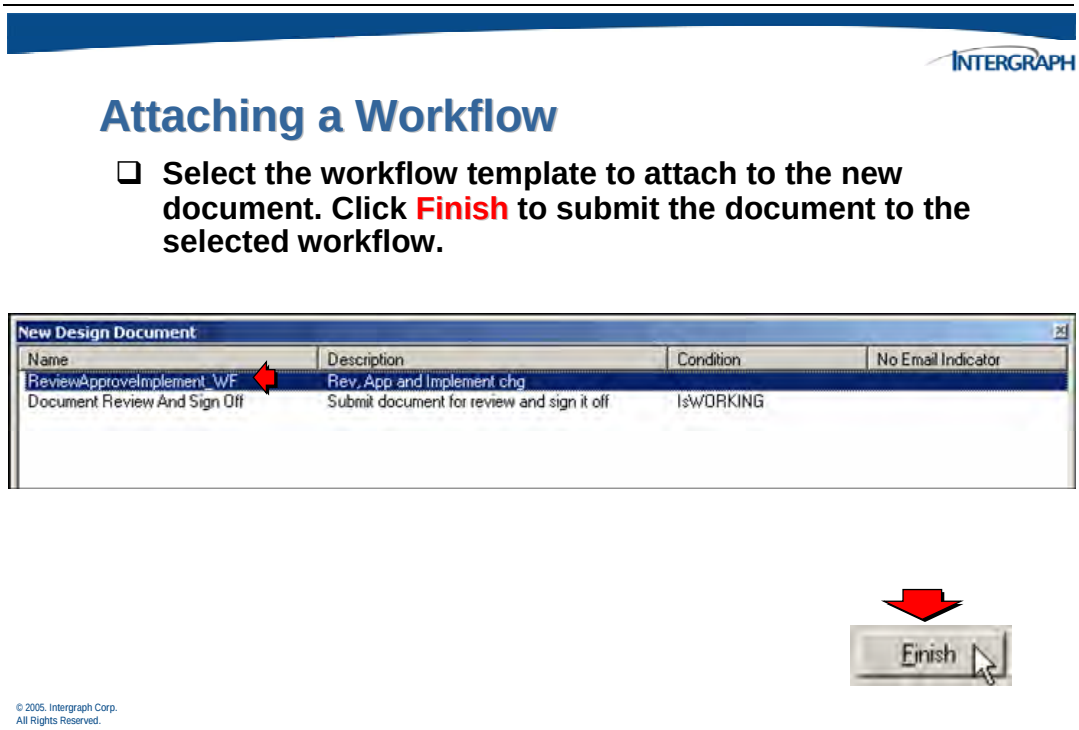




© 2005. Intergraph Corp.
All Rights Reserved.

7.8.1 Attaching an Object to a Workflow

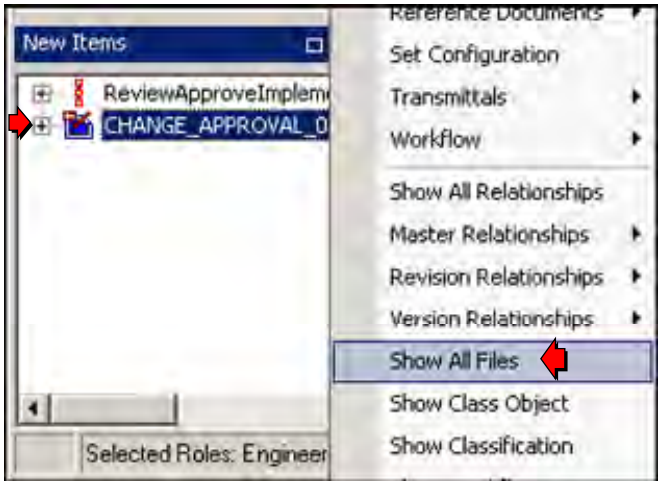
In order to submit the document and attached file to the workflow, the workflow must first be attached to the document. To start the workflow review and approval process, continue with the following steps:



Attaching the workflow will also submit the document to the workflow. Verify that the file was successfully attached to the document.

Verifying Attached Files

- **Right-click on the new *document* and choose *Show Files* from the dynamic menu.**

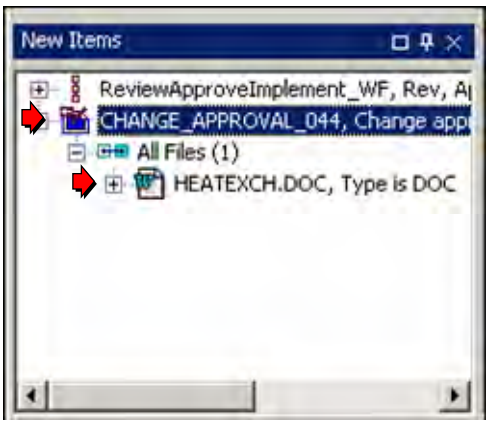


© 2005. Intergraph Corp.
All Rights Reserved.

The *New Items* window will expand to show the attached file.

Verifying Attached Files

- **Verify that a file is attached to the selected document.**



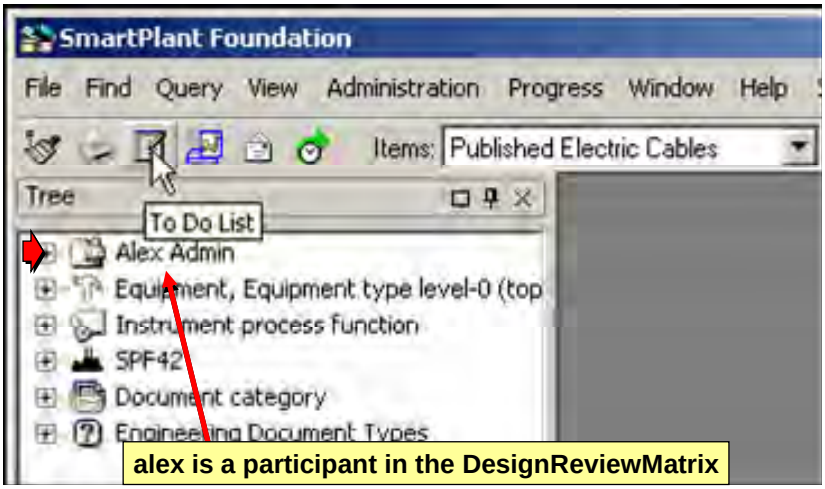
© 2005. Intergraph Corp.
All Rights Reserved.

7.8.2 Viewing an Active Workflow

Once an object has been submitted to the workflow, the active workflow can be displayed graphically. The first recipient in the list for the selected workflow, *ReviewApproveImplement_WF*, is the **DesignReviewMatrix** document matrix. That means that any participant of that matrix will receive a task in their **To Do List** as well as an email notification.

Viewing Workflow Progress

- **From the toolbar, choose the *To Do List* icon.**

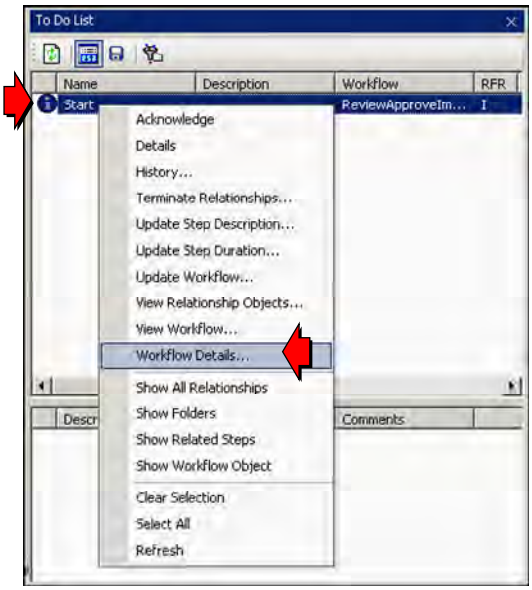


© 2005. Intergraph Corp.
All Rights Reserved.

The **To Do List** will show the first step in the workflow, *Start*.

Viewing Workflow Progress

- **Right-click on the Start step in the To Do List window and choose *Workflow Details* from the menu.**

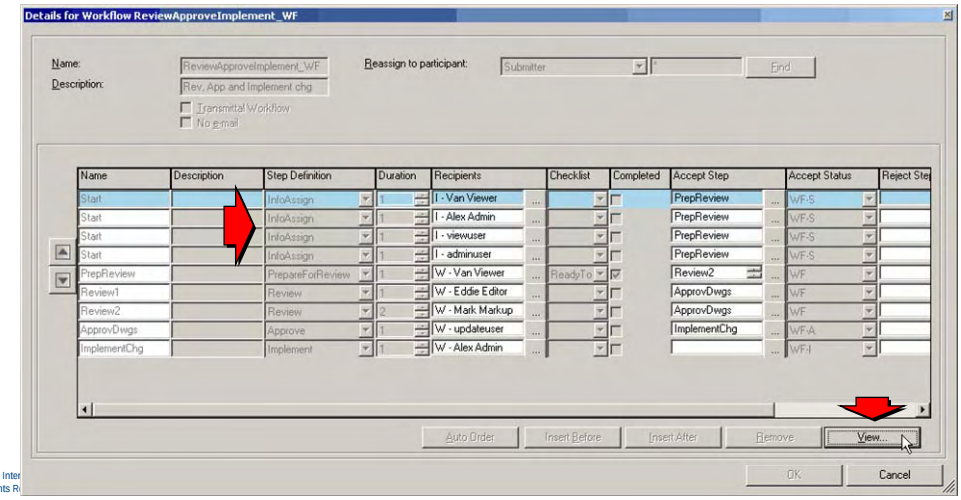


© 2005. Intergraph Corp.
All Rights Reserved.

The *Workflow Details* form for the **ReviewApproveImplement_WF** workflow.

Viewing Workflow Progress

- **Click on the View button to see a graphical representation of the workflow progress.**

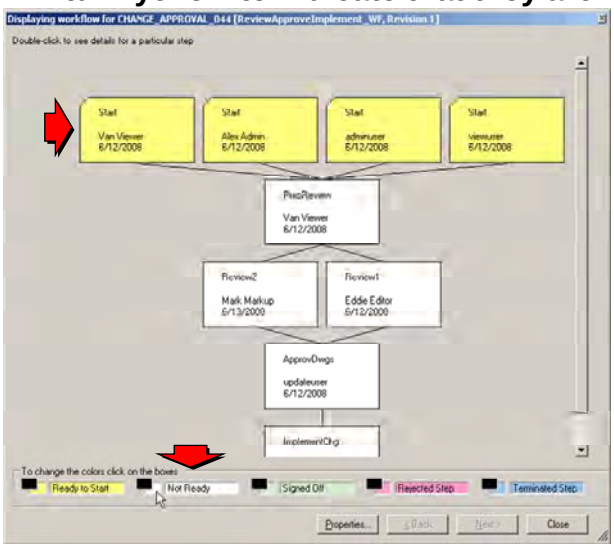


© 2005. Inter
All Rights R

A graphical workflow dialog will be displayed.

Viewing Workflow Progress

- **The ready steps will turn yellow to indicate that they are ready to start.**



© 2005. Intergraph Corp.
All Rights Reserved.

You can configure the display color of any of the workflow steps in the graphical display.

Viewing Workflow Progress

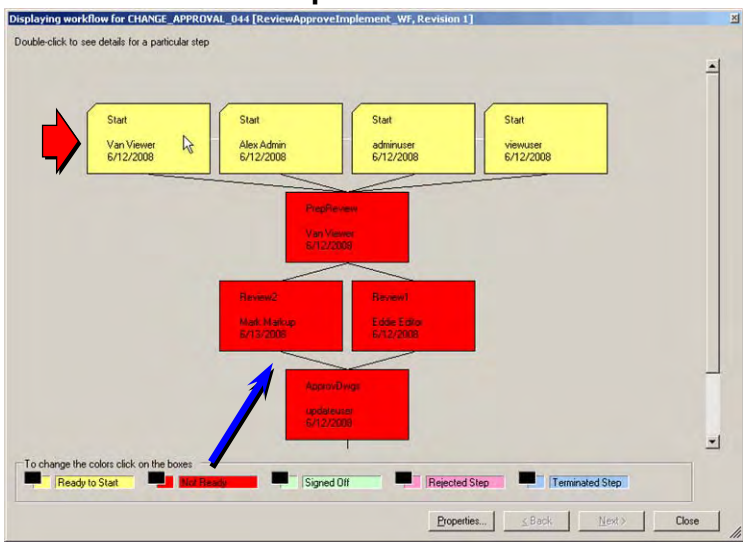
- **Select a new color for the step display.**

© 2005. Intergraph Corp.
All Rights Reserved.

The *Not Ready* steps are now displayed in red. A step's details can also be reviewed.

Viewing Workflow Progress

- **Double-click on a step to view it's details.**

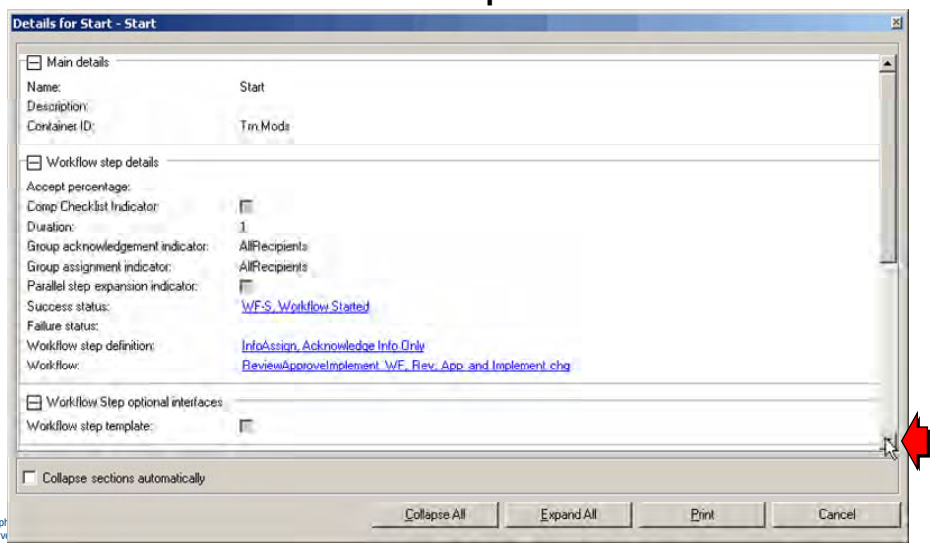


© 2005. Intergraph Corp.
All Rights Reserved.

A *Step Details* dialog window will display.

Viewing Workflow Progress

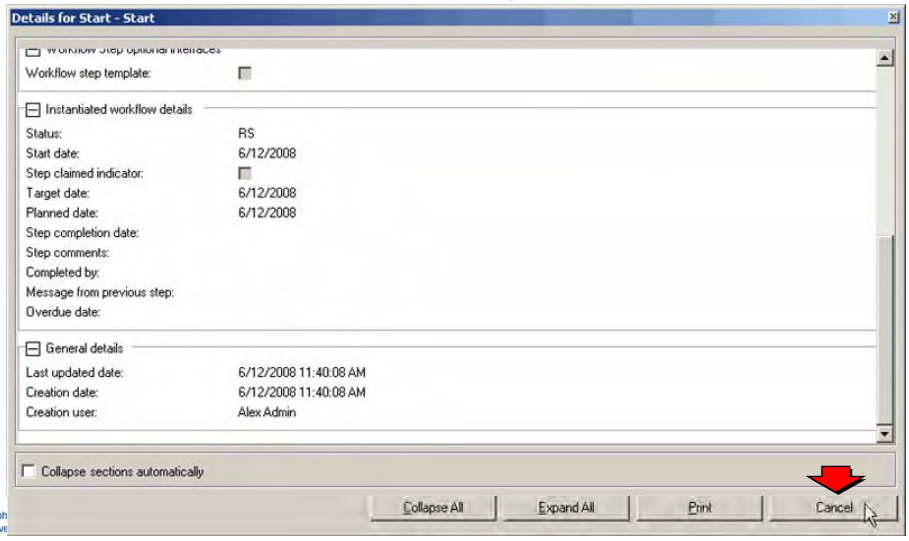
- **Scroll down to view the step details.**



© 2005. Intergraph
All Rights Reserv

Viewing Workflow Progress

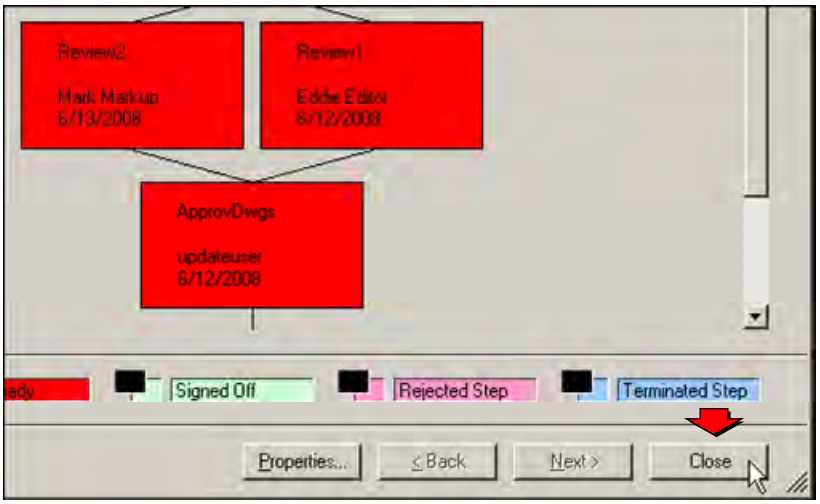
- **Click Cancel to close the *Step Details* dialog.**



© 2005. Intergraph
All Rights Reserve

Viewing Workflow Progress

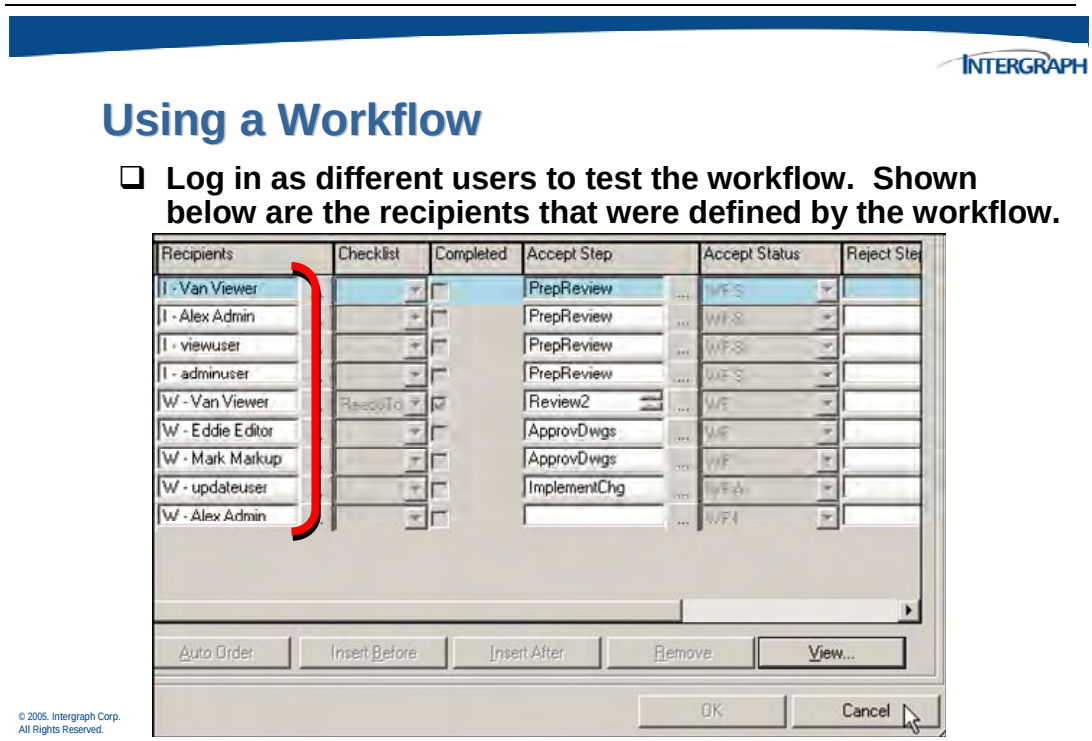
- **Click on the Close button to close the graphical display.**



© 2005. Intergraph Corp.
All Rights Reserved.

7.8.3 Workflow Signoffs

A workflow is made up of different steps with one or more users assigned to perform some kind of task within each step.

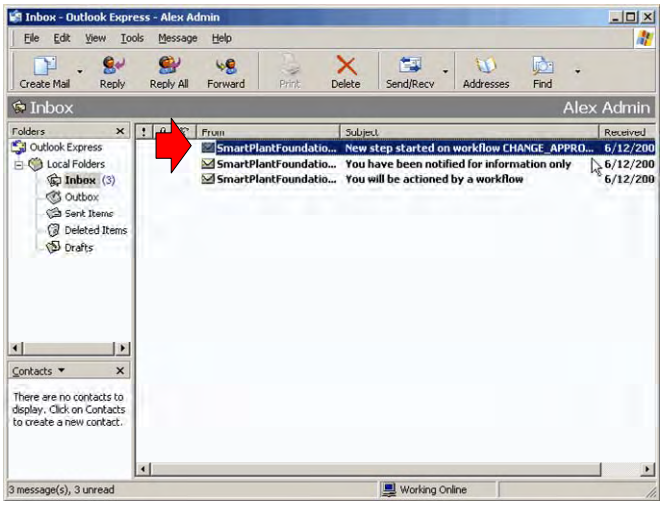


The recipient for the first step in the workflow will log in to the SPF Desktop client and check the **To Do List**. Using the information about the workflow from the previous pages, this recipient would be a participant in the *DesignReviewMatrix* document matrix.

Submitting a document to a workflow will also send out an email notification. You may have to switch the email client to the correct user. The first user for this particular workflow will be **Alex Admin**.

Workflow Email Notification

- **Double-click to view the mail message sent by SmartPlant Foundation.**

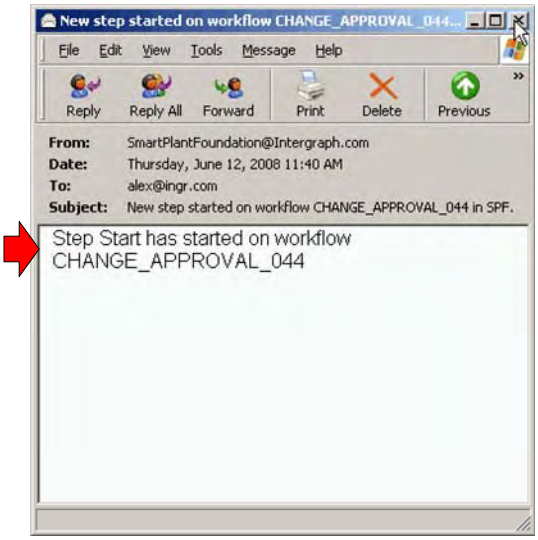


© 2005. Intergraph Corp.
All Rights Reserved.

The selected mail message will be opened.

Workflow Email Notification

The required action is detailed in the mail message.

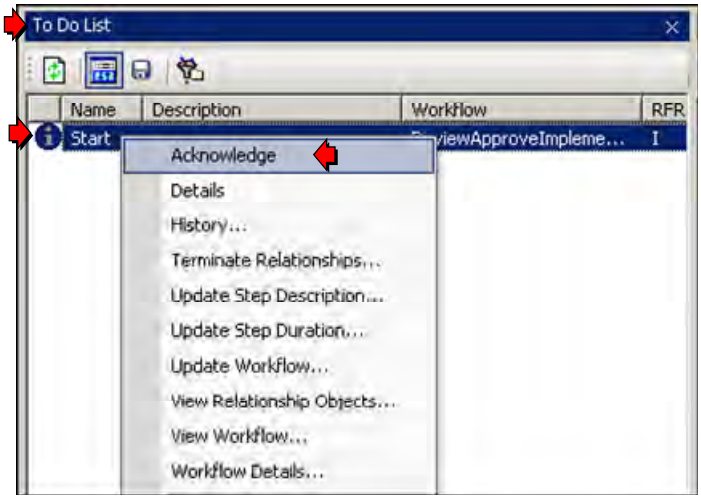


© 2005. Intergraph Corp.
All Rights Reserved.

Since user **alex** created the new document and submitted it to the workflow, which is the user that will respond to the ***Start*** step in the *To Do List* by signing off or **Acknowledging** this step.

Workflow Step Sign Off

- **In the To Do List window, right-click on the step (InfoAssign) and select *Acknowledge* from the menu.**

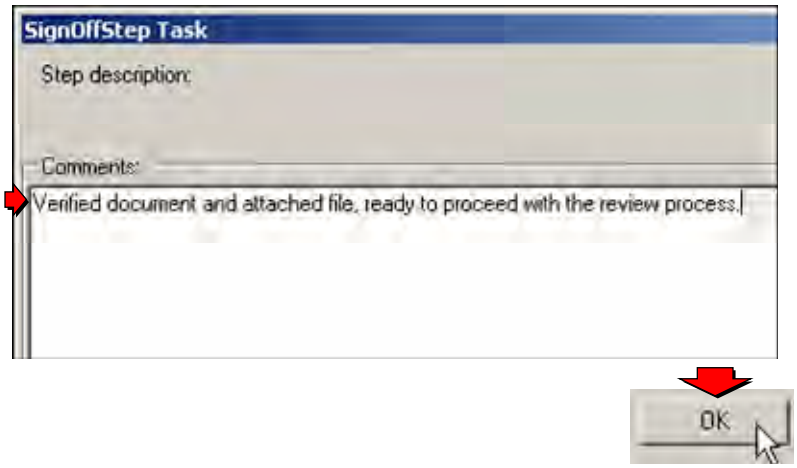


© 2005. Intergraph Corp.
All Rights Reserved.

A *SignoffStep Task* dialog will display that will allow the workflow to capture optional comments as this step is *Acknowledged* by the recipient.

Workflow Step Sign Off

- **Enter an optional comment and click OK.**

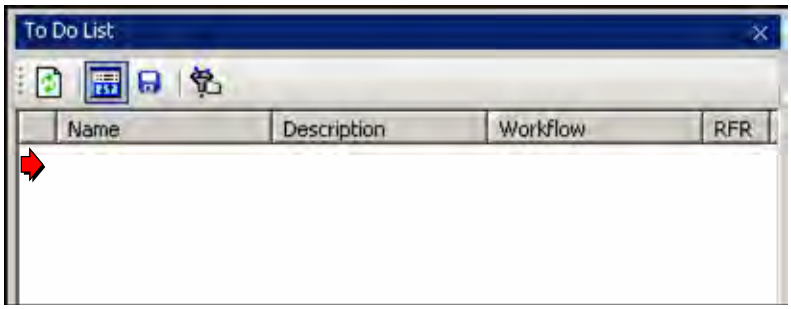


© 2005. Intergraph Corp.
All Rights Reserved.

Once the step has been signed off or *Acknowledged*, the recipient no longer has any access to the step and it is removed from the *To Do List*.

Workflow Step Sign Off

- **The step will disappear from the To Do List window.**

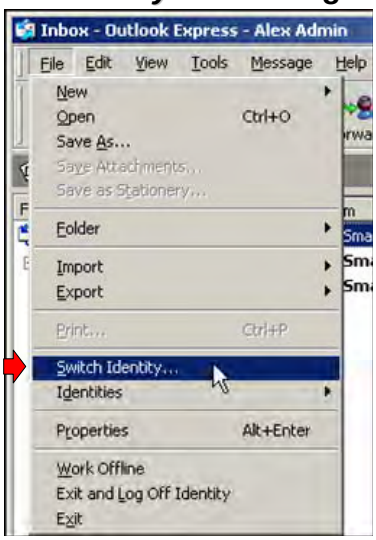


© 2005. Intergraph Corp.
All Rights Reserved.

Restore the email window and change the email account to the next user in this workflow.

Workflow Email Notification

- **Select *File > Switch Identity…* to change the email account being accessed.**



© 2005. Intergraph Corp.
All Rights Reserved.

Use the email account **Van Viewer** to review any notifications sent by SPF.

Workflow Email Notification

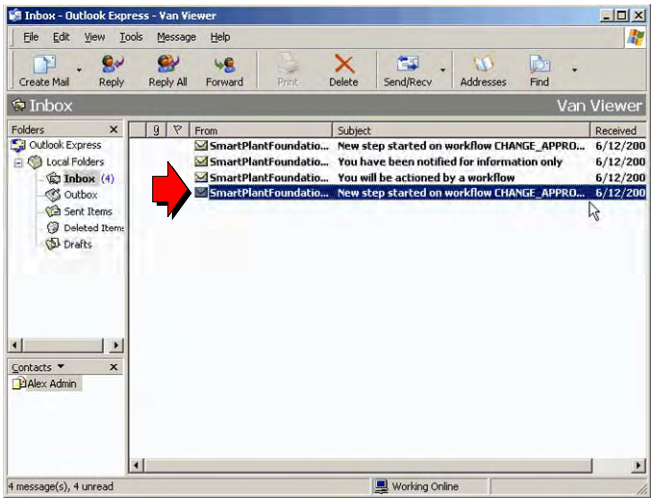
- **From the list, select *Van Viewer* and click OK.**

© 2005. Intergraph Corp.
All Rights Reserved.

The email *Inbox* for user Van Viewer will display.

Workflow Email Notification

- **Double-click to view the mail message sent by SmartPlant Foundation.**

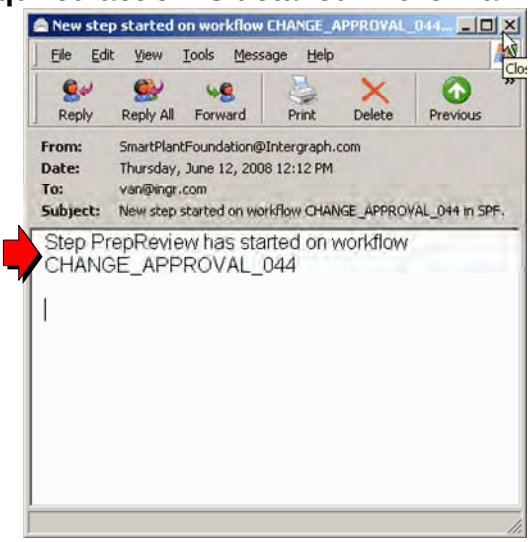


© 2005. Intergraph Corp.
All Rights Reserved.

The selected mail message will be opened.

Workflow Email Notification

- **The required action is detailed in the mail message.**

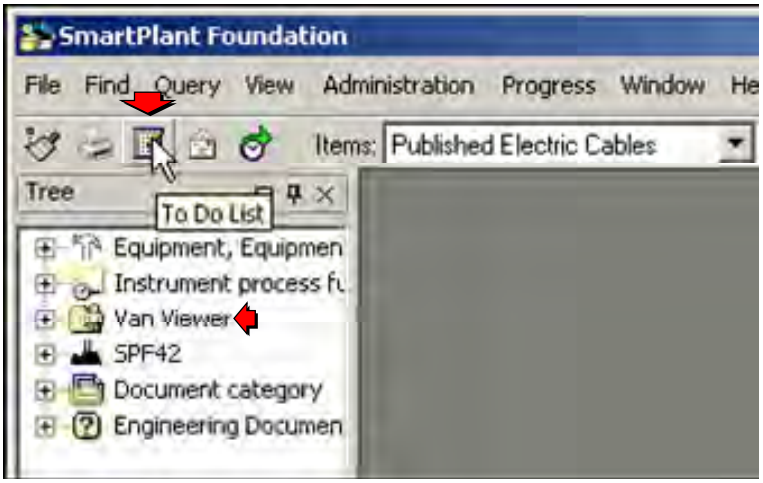


© 2005. Intergraph Corp.
All Rights Reserved.

The next recipient in the workflow process will log in to the SPF client. The first action would be to check the *To Do List* for any workflow assignments.

Workflow Step Sign Off

- **From the toolbar, choose the *To Do List* icon.**

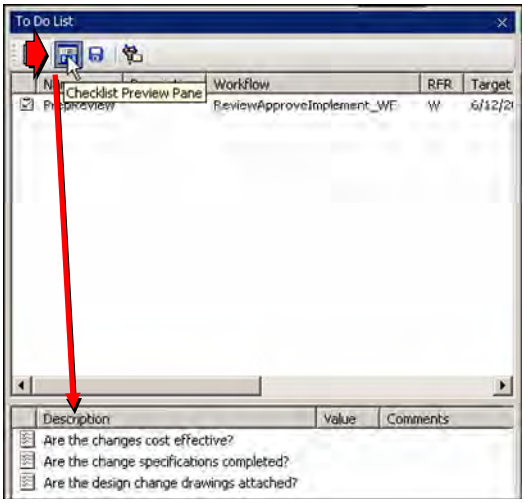


© 2005. Intergraph Corp.
All Rights Reserved.

Remember, checklists are a series of tasks or questions that the user can be required to complete before a step can be completed in a workflow.

Completing a Workflow Checklist

- **Click the *Checklist Preview Pane* icon in the To Do List to display any check lists associated with this step.**

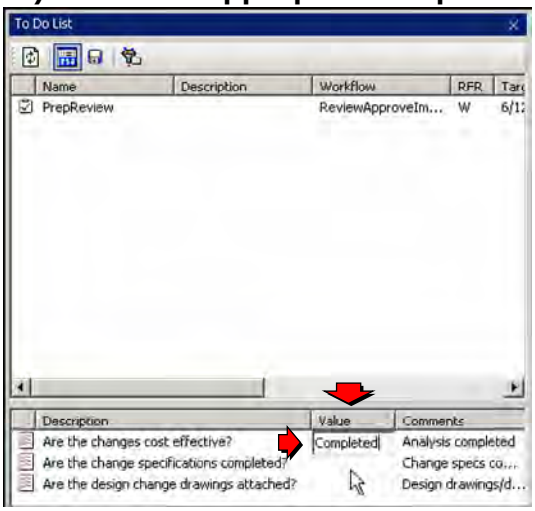


© 2005. Intergraph Corp.
All Rights Reserved.

In this example, the user will complete the checklist questions before continuing.

Completing a Workflow Checklist

- **Click in the Value field (to the right of the check list sequence) and enter appropriate responses.**

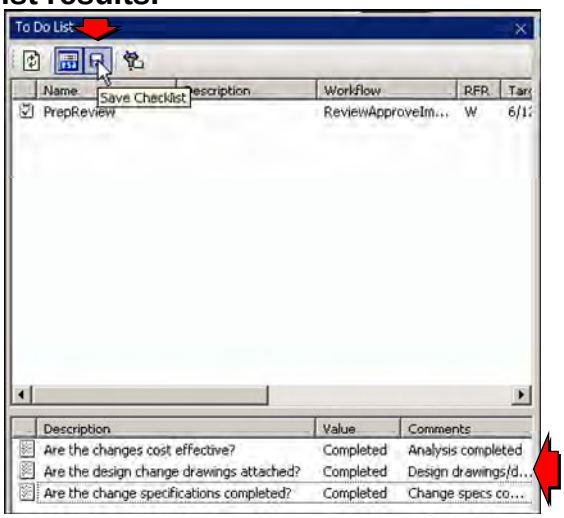


© 2005. Intergraph Corp.
All Rights Reserved.

Save the results of the completing the checklist.

Completing a Workflow Checklist

- **Click the *Save Checklist* icon in the To Do List to save check list results.**

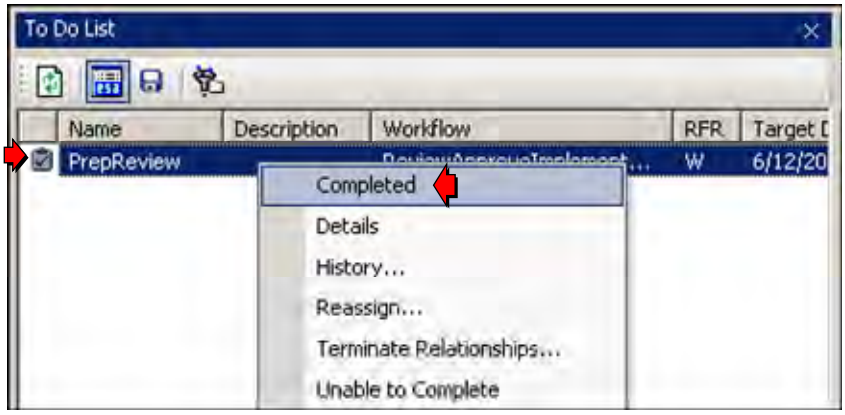


© 2005. Intergraph Corp.
All Rights Reserved.

Once the checklist questions have been answered, the step (*PrepReview*) can be signed off. Since the check list was a type of task, the sign off response is **Completed**.

Workflow Step Sign Off

- **In the To Do List window, right-click on the step (PrepReview) and select *Completed* from the menu.**

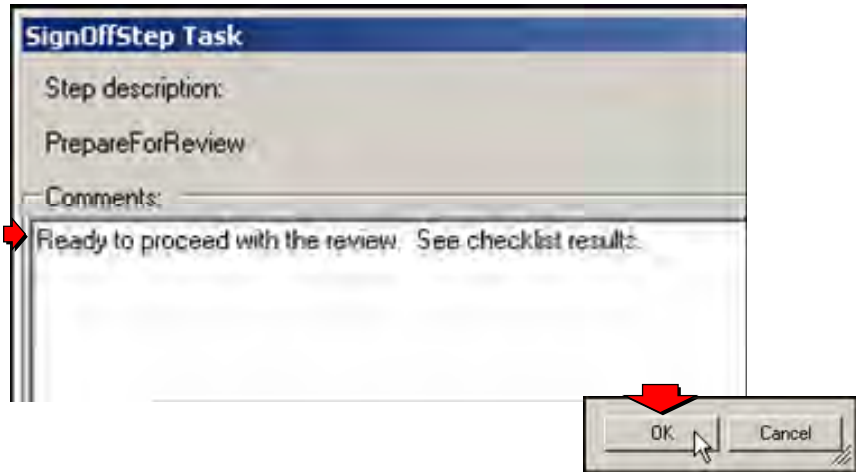


© 2005. Intergraph Corp.
All Rights Reserved.

A *SignoffStep Task* dialog will display that will again allow the workflow to capture comments as this step is *Completed* by the recipient.

Workflow Step Sign Off

- **Enter an optional comment and click OK to complete the step signoff.**

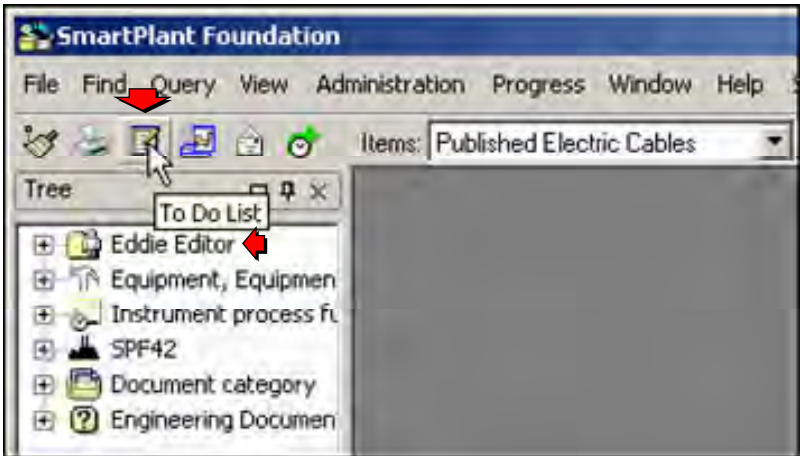


© 2005. Intergraph Corp.
All Rights Reserved.

The next recipient in the workflow process will log in to the SPF client. Again, the first action should be to check the *To Do List* for any workflow assignments.

Workflow File Viewing

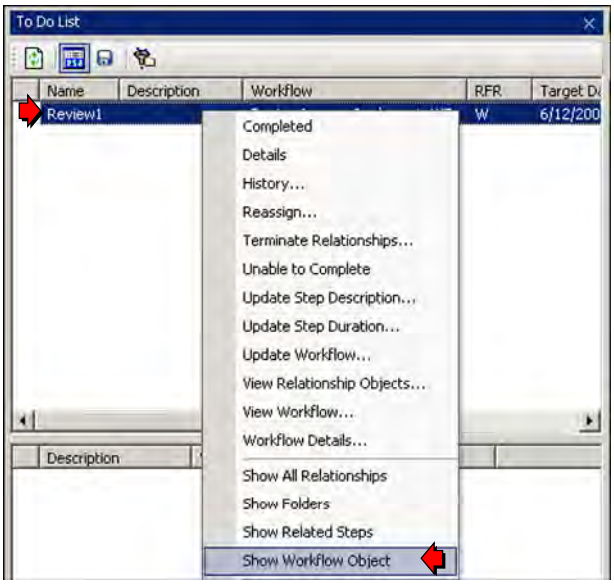
- **From the toolbar, choose the *To Do List* icon.**



© 2005. Intergraph Corp.
All Rights Reserved.

As part of the review process, the user may need to view the attached graphical file.

Workflow File Viewing



- **In the To Do List window, right-click on the step (Review1) and select *Show Workflow Object* from the menu.**

The CHANGE_APPROVAL_044 document object will be displayed in a browser window.

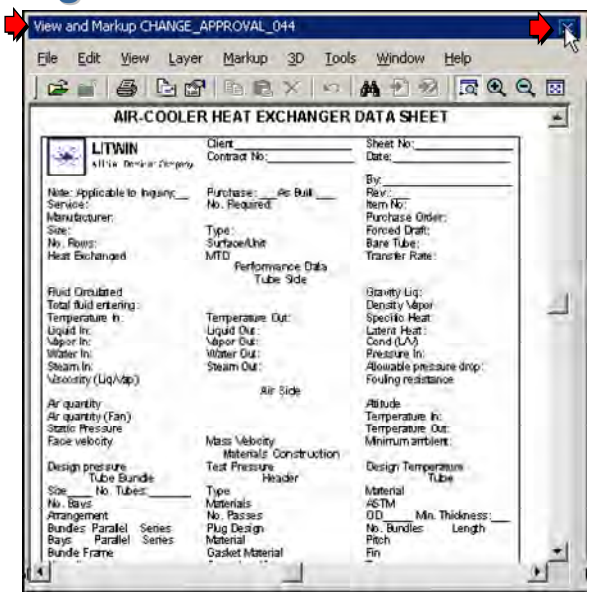
Workflow File Viewing

- **Right-click on the workflow object (CHANGE_APPROVAL_044) and select *Files > View and Markup*.**

The attached file will display in a *View and Markup* window.

Workflow File Viewing

- **Click on the X in the upper right corner of the window to close *View and Markup*.**

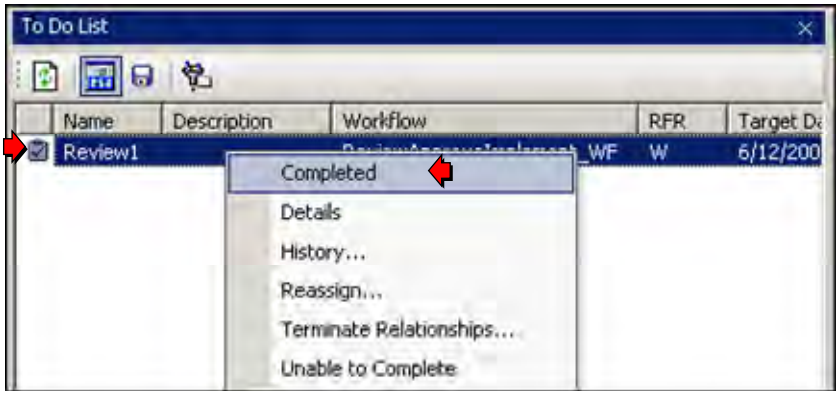


© 2005. Intergraph Corp.
All Rights Reserved.

After the *File Viewer* has been closed, complete the sign off of this workflow step.

Workflow Step Sign Off

- **In the To Do List window, right-click on the step (Review1) and select *Completed* from the menu.**

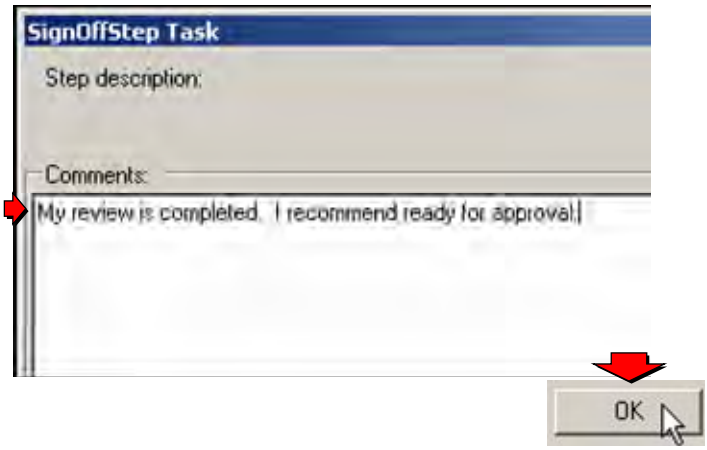


© 2005. Intergraph Corp.
All Rights Reserved.

A *SignoffStep Task* dialog will display so that this step can be *Completed* by the recipient.

Workflow Step Sign Off

- **Enter an optional comment and click OK to complete the step signoff.**

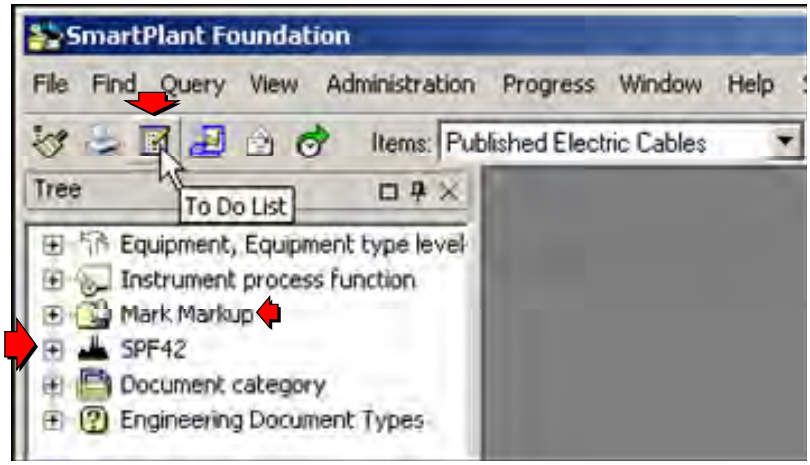


© 2005. Intergraph Corp.
All Rights Reserved.

This step happens to be a parallel step where two (or more) people can review the object at the same time but in separate steps. These steps could be configured so that the first person to signoff on the step, causes the workflow to continue.

Workflow File Viewing

- **Again, after setting the Scope, choose the *To Do List* icon.**

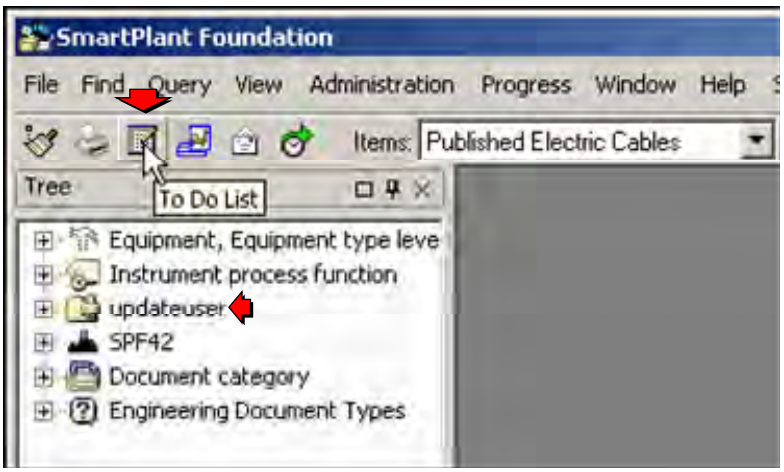


© 2005. Intergraph Corp.
All Rights Reserved.

The next recipient in the workflow process will log in to the SPF client.

Viewing Workflow Progress

- **From the toolbar, choose the *To Do List* icon.**

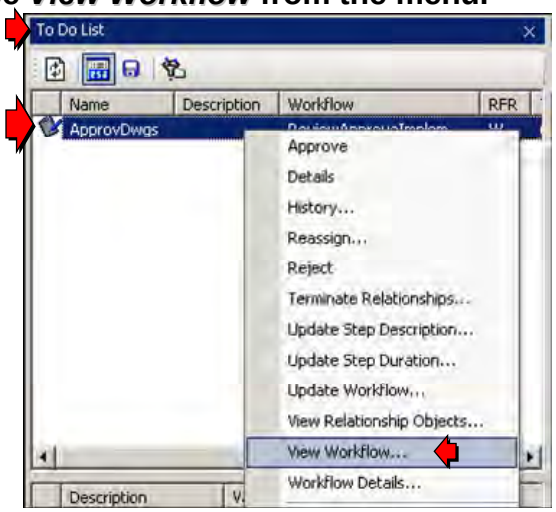


© 2005. Intergraph Corp.
All Rights Reserved.

A graphical view of the workflow progress can be displayed from the workflow step.

Viewing Workflow Progress

- **Right-click on the step in the To Do List window and choose *View Workflow* from the menu.**

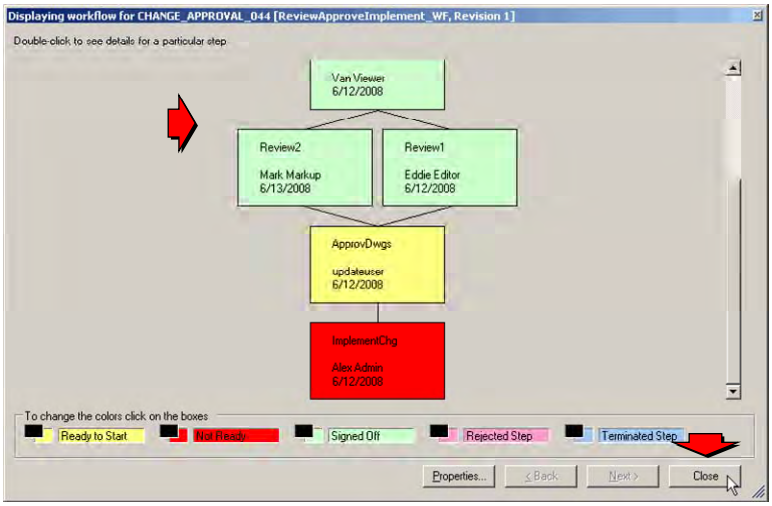


© 2005. Intergraph Corp.
All Rights Reserved.

In this example, ***updateuser*** is the user checking the workflow progress.

Viewing Workflow Progress

- **The signed off steps will turn green to indicate that they are completed.**

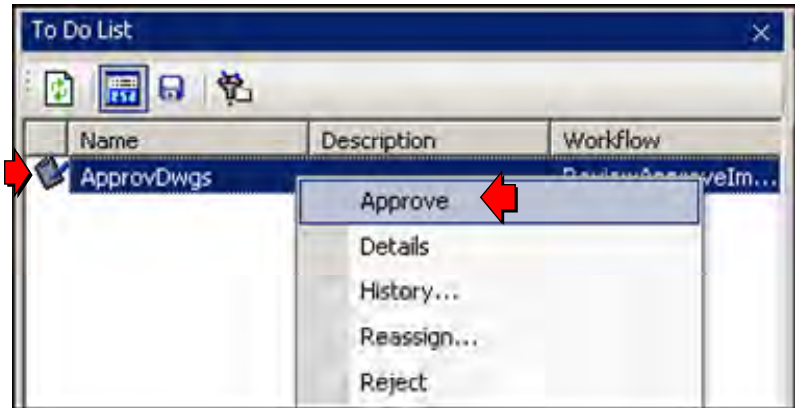


© 2005. Intergraph Corp.
All Rights Reserved.

This is an **Approve** step for a specific recipient.

Workflow Step Sign Off

- **In the To Do List window, right-click on the step (ApprovDwgs) and select *Approve* from the menu.**

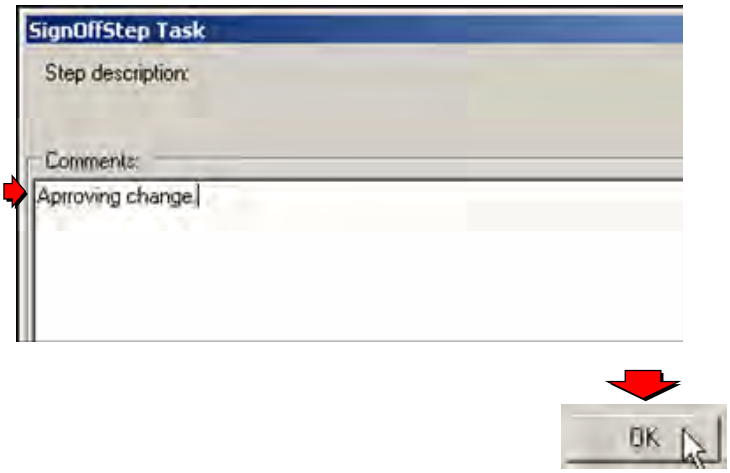


© 2005. Intergraph Corp.
All Rights Reserved.

A *SignoffStep Task* dialog will display so that this step can be *Approved* by the recipient.

Workflow Step Sign Off

- **Enter an optional comment and click OK.**

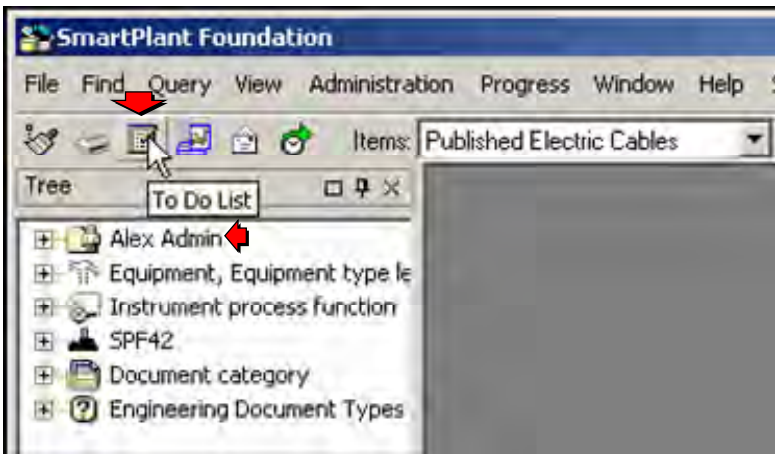


© 2005. Intergraph Corp.
All Rights Reserved.

Perform the sign off for the last step in the workflow.

Workflow Step Sign Off

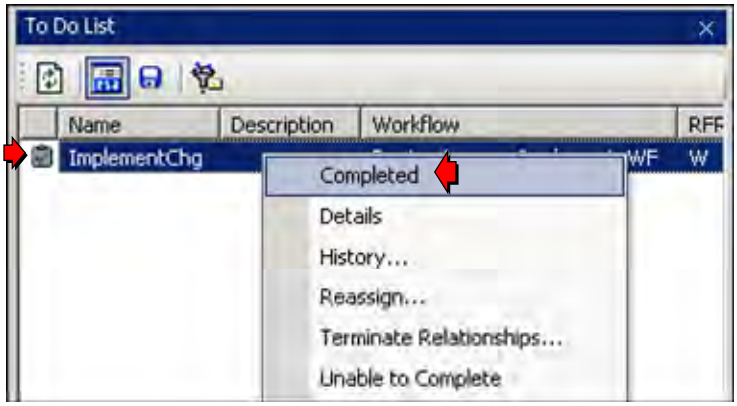
- **From the toolbar, choose the *To Do List* icon.**



© 2005. Intergraph Corp.
All Rights Reserved.

Workflow Step Sign Off

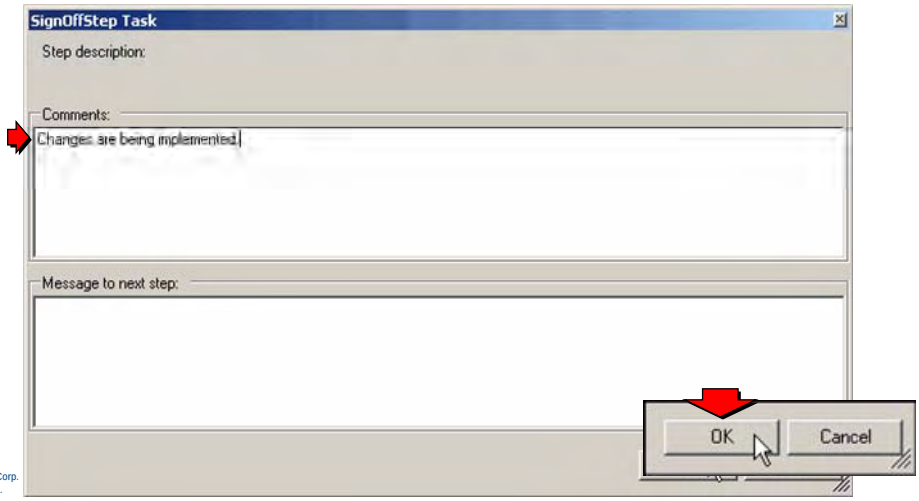
- **In the To Do List window, right-click on the step (ImplementChg) and select *Completed* from the menu.**



© 2005. Intergraph Corp.
All Rights Reserved.

Workflow Step Sign Off

- **Enter an optional comment and click OK to complete the step signoff.**



© 2005. Intergraph Corp.
All Rights Reserved.

Since this is the last step in this workflow example, there are no more steps to be processed. The workflow is now completed.

7.9 Workflow History

A ***Show Workflow*** command can be used to once again bring up the workflow graphical display. First, perform a **Find** to locate a document that has completed a workflow.

Workflow History

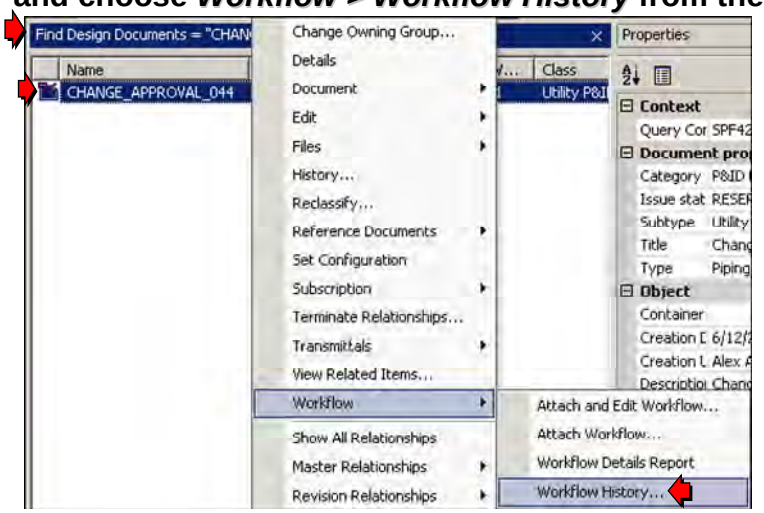
- **Enter any desired search criteria and click OK to find the Design Document workflow object.**

© 2005. Intergraph Corp.
All Rights Reserved.

A *Find Results* list view window will be displayed.

Workflow History

- **Right-click on the document in the Find results window and choose *Workflow > Workflow History* from the menu.**

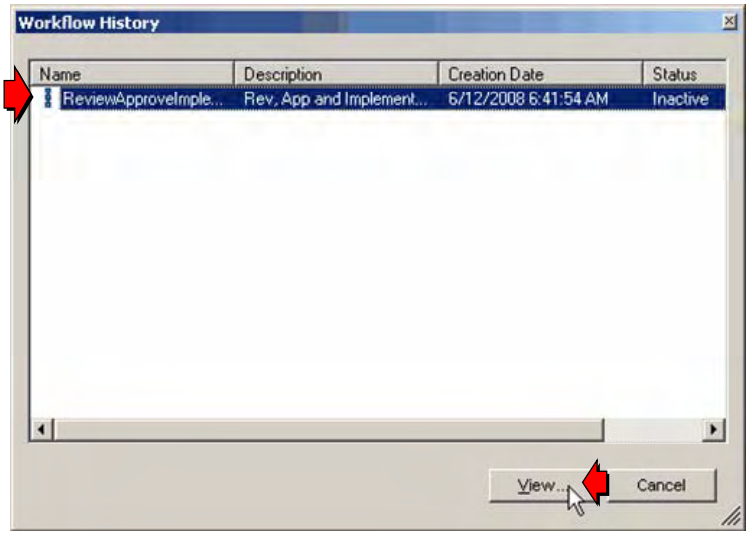


© 2005. Intergraph Corp.
All Rights Reserved.

A *Workflow History* form will appear.

Workflow History

- **Select the displayed workflow and choose the *View* button.**



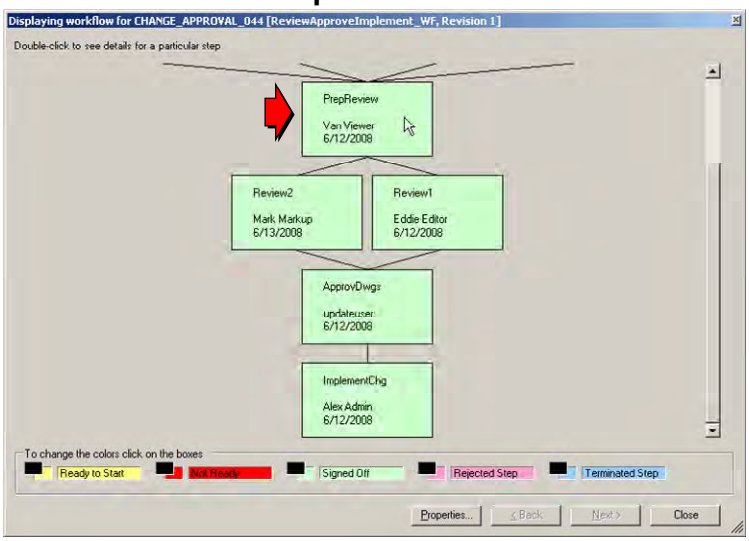
© 2005. Intergraph Corp.
All Rights Reserved.

Since this instance of the workflow has completed, note the **Inactive** *Status*.

Note that all steps will show the color green indicating successful completion.

Workflow History

- **Double-click on a step to view it's details.**

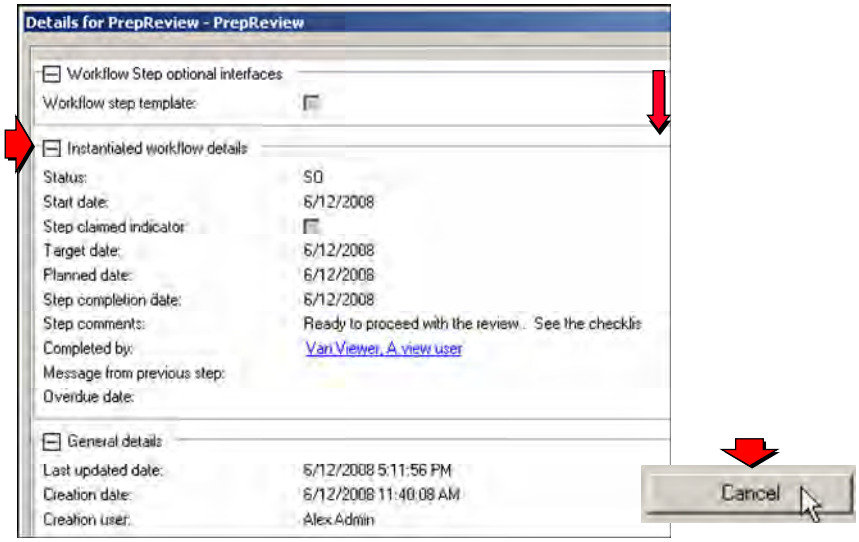


© 2005. Intergraph Corp.
All Rights Reserved.

PrepReview is the step that was associated with a checklist.

Workflow History

- **Choose Cancel to dismiss the *Details* form.**

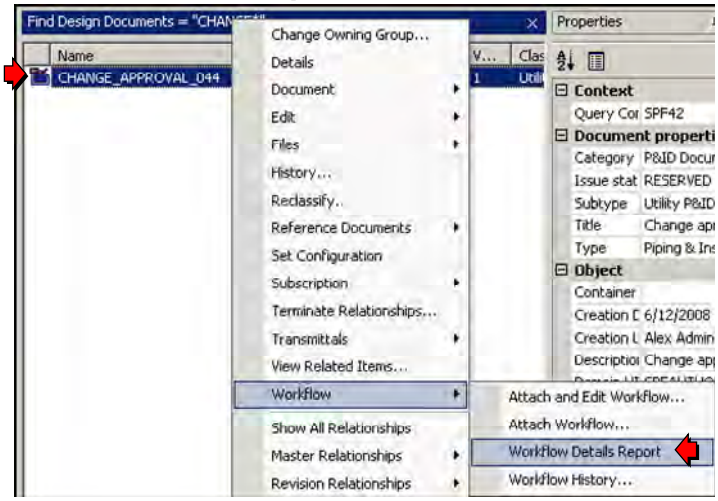


© 2005. Intergraph Corp.
All Rights Reserved.

A *Workflow Details Report* can be generated to display a history of the workflow processing.

Workflow History

- **Right-click on the document and select *Workflow* > *Workflow Details Report* from the menu.**

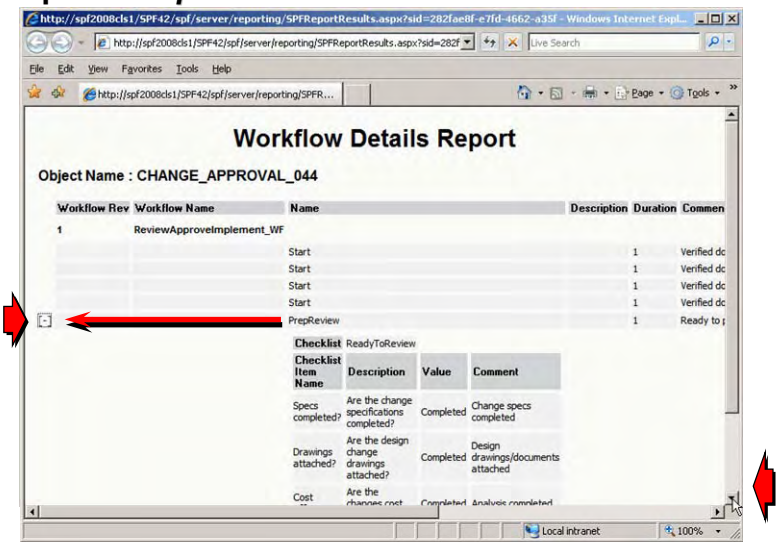


© 2005. Intergraph Corp.
All Rights Reserved.

This report will display details about the finished workflow such as the *Name*, *Step Names*, and *Assignees*.

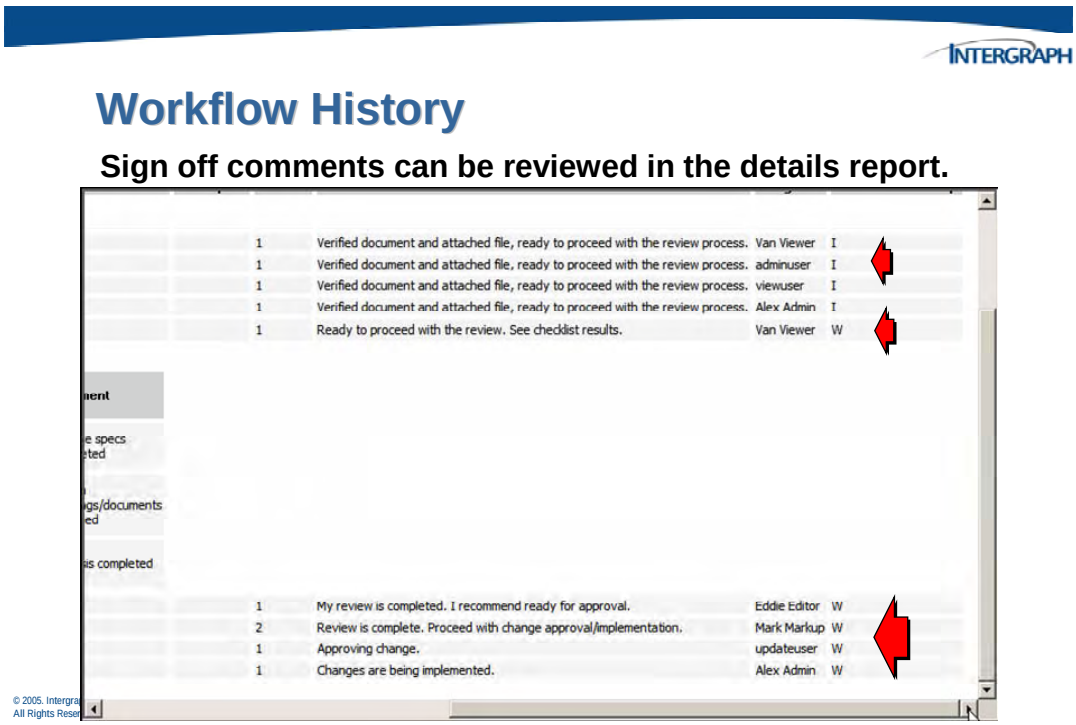
Workflow History

- **Expand *PrepReview* to view the Checklist comments.**



© 2005. Intergraph Corp.
All Rights Reserved.

Comments made by recipients during sign off are displayed in the report. Use the scroll bar to review the details report.



DSPF1-TP-100016A

SmartPlant Foundation 2008 (4.2) Configuration and Administration II

Course Guide **Volume 1**