SIEMENS

Teamcenter 11.1 Systems Engineering and Requirements Management

Systems Architect/ Requirements Management Project Administrator's Manual

REQ00002 • U

Manual History

Manual Revision	Teamcenter Requirements Version	Publication Date
A	3.0	March 2003
B	3.0.1	May 2003
C	4.0	December 2003
D	4.1	February 2004
E	5.0	July 2004
F	6.0	March 2005

Manual Revision	Teamcenter Systems Engineering and Requirements Management Version	Publication Date
G	2005	September 2005
H	2005 SR1	June 2006
I	2007	December 2006
J	2007.1	April 2007
K	2007.2	September 2007
L	2007.3	January 2008
M	8	January 2009
N	8.0.1	June 2009
O	8.1	October 2009
P	8.2	October 2010
Q	9	July 2011
R	9.1	May 2012
R1	9.1.5	January 2014

Manual Revision	Teamcenter Systems Engineering and Requirements Management Version	Publication Date
S	10.0	January 2015
T	10.1	September 2016
U	11.1	March 2018

This edition obsoletes all previous editions.

Contents

Contents

Preface

This manual is a project administrator's reference for Teamcenter Systems Architect/Requirements Management 11.1. Systems Architect/Requirements Management belongs to the Siemens PLM Software portfolio of digital product lifecycle management software and services.

Audience

This manual is for Systems Architect/Requirements Management (Architect/Requirements) project administrators. The manual provides both conceptual information and step-by-step instructions for specific tasks.

This manual assumes that you are familiar with your project and your product development process, that you understand general computer terminology and the Microsoft Windows operating system, and that you have experience with Microsoft Word, Microsoft Excel, and Microsoft Office Visio.

Project Administrator privilege is required for all procedures in this manual.

Conventions

This manual uses the conventions described in the following sections:

Revision Marks

Technical changes are marked by a bar adjacent to the changed text.

Browser and Dialog Window Examples

The examples of browsers and dialog windows in this manual may appear different from those you see on your screen:

- The examples reflect Systems Architect/Requirements Management as initially installed at your site. Your enterprise may customize the browsers and dialog windows such that they appear different from those in the examples.
- The examples reflect individual Systems Architect/Requirements Management modules. If you install additional modules, your dialog windows and browsers reflect the additional modules.

- The examples reflect Systems Architect/Requirements Management installed on a Windows platform.

Names and Values

This manual represents system names, file names, and values in fonts that help you interpret the name or value. For example:

Change or add the parameter to the **init***sid***.ora** file.

The conventions are:

Bold	Bold font represents unvarying text or numbers within a name or value. Capitalization is as it appears.
Italic	Italic font represents text or numbers that vary within a name or value. The characters in italic text describe the entry. Letters are shown in lowercase, but the varying text may include uppercase letters. In **init***sid***.ora**, *sid* identifies a varying portion of the name (a unique system ID). For example, the name of the file might be: **initBlue5.ora**
text-text	A hyphen separates two words that describe a single entry.

Submitting Comments

Portions of Teamcenter software are provided by third-party vendors. Special agreements with these vendors require Siemens PLM Software to handle all problem reports concerning the software they provide. Please submit all comments directly to Siemens PLM Software.

Please feel free to share with us your opinion on the usability of this manual, to suggest specific improvements, and to report errors. Mail your comments to:

Siemens PLM Software Technical Communications
5939 Rice Creek Parkway
Shoreview, MN 55126
U.S.A.

To submit an incident report, you can use the Siemens PLM Software GTAC online support tools at the following URL:

http://www.plm.automation.siemens.com/en_us/support/gtac/

Proprietary and Restricted Rights Notice

Chapter 1: Introduction to Project Administration

This chapter presents an overview of projects in Architect/Requirements, describes the types of schema objects used in the Administration module, and contains instructions for creating and importing projects, for exporting and importing project data, and for deleting schema objects and entire projects.

Enterprise Administrator privilege is required for creating a project and importing a project. **Project Administrator** privilege is required for all other procedures in this manual.

Overview of Projects

In Architect/Requirements, a project defines logical boundaries for the following:

- Access privileges of individual users

  Access privileges determine which users can modify the information in that project, and which can only view the information. Each user can be granted access to any number of projects, with different permissions for each project.

  Users can be granted project access privileges at several broad levels, with corresponding restrictions on all user actions within each project. This is in addition to the permissions imposed through the security profile settings on individual objects within a project.

- Customization of object properties and object types

  At project scope, an Architect/Requirements project administrator defines custom object properties, property values, and object subtypes. Those properties, values, and subtypes apply only within a single project. Therefore, each project can be customized for a particular purpose, process, or product.

Schema Objects

In the Architect/Requirements Administration module, project administrators work with the following types of schema objects:

- *Property definitions*
- *Type definitions*
- *Templates and style sheets*
- *Reports*
- *Views*
- *Users and user groups*
- *Security profiles*
- *Activators*

Property Definitions

A project administrator can create custom property definitions to tailor Architect/Requirements for an organization's requirements management process, product types, and business needs. Each property is defined once, and can then be applied to as many different object types as desired.

For each property, the property definition determines the following:

- The property type (*choice*, *date*, *numeric*, or *text*)
- The default value for that property to be assigned to newly created objects.

For choice properties, the property definition also determines the list of valid choices, and whether the list should have single-choice or multiple-choice behavior.

Property definitions are defined within the scope of a project, and can be applied only to objects within that project. Different projects may have properties with the same names, but the properties are entirely independent.

Type Definitions

A type definition specifies the properties that apply to all objects of a given type. In turn, these properties determine the nature and behavior of the related objects.

Type definitions also allow the project administrator to define custom object types, which are subtypes based on the built-in object types. For example, it may be useful to define several subtypes of requirements, each with unique identifying properties:

- Customer Requirement — CustomerID property
- Mil Spec — Requirements extracted from DoD specifications, with a DoD Spec Number property
- Company Policy — Standard Procedure Document Number property

Type definitions are defined within the scope of a project and can only be applied to objects within that project. Different projects may have object types with the same names, but they are entirely independent of each other.

Templates and Style Sheets

When users export object data from the Systems Engineering and Requirements Management module to Microsoft Word, Architect/Requirements references a *document template* to generate the export document. In turn, the document template supplies instructions through the *object templates* and the *style sheet* that are associated with the document template.

A document template, two object templates, and a style sheet are created automatically for every new project. In the object templates, *tags* represent the object properties whose values are extracted to the export document for each object that the user specifies. The style sheet determines the Microsoft Word formatting that is applied to the data in the export document.

The project administrator can modify the built-in document template, object templates, and style sheet to customize the export process. In addition, the project administrator can create custom document templates, object templates, and style sheets according to specific project needs.

Reports

Architect/Requirements provides built-in reports through the optional **Standard Reports** package, which the project administrator can add to any project. In addition, users can create and save reports based on search criteria entered in the Search module.

Standard and custom reports are project specific. All reports for a project are stored in the **Reports and Formatting** folder in the Administration module. For more information about using the Search module and reports, see the *Systems Architect/Requirements Management User's Manual*.

The following reports are available in the **Standard Reports** package:

- **Orphan Requirements** shows requirements that have no defining trace links. The search starts at the selected project node, folder, or requirement. All lower level requirements that match the search criteria are shown in the Search Results dialog window.

- **Requirement Approval Status** shows requirements whose **Approval Status** property value is **Change Approved**. The search starts at the selected project node, folder, or requirement. All lower level requirements that match the search criteria are shown in the Search Results dialog window.

  The **Change Approved** value is set when the change approval package receives an **Approved** event for the submitted requirement. For more information, see Change Approval, in chapter *Configuring the Change Management Package*.

- **Show Impact Downstream** shows all objects in the complying path from the selected object. These complying objects are affected if the selected object is changed. This report is output to the Search Results dialog window. If the project node is selected, the report shows only the project node.

- **Show Impact Upstream** shows all objects in the defining path to the selected object. These defining objects are affected if the selected object is changed. This report is output to the Search Results dialog window. If the project node is selected, the report shows only the project node.

-

Trace Report Deep shows complying trace links for all objects at all levels in the selected folder. This report is output to the Search Results dialog window.

- **Trace Report Deep Table Format** shows complying trace links for all objects at all levels in the selected folder. This report is output to a Microsoft Excel spreadsheet.

- **Unallocated Requirements** shows requirements that have no complying trace links. The search starts at the selected project node, folder, or requirement. All lower level requirements that match the search criteria are shown in the Search Results dialog window.

To add the standard reports to a project:

1. In the navigation tree, select the **Projects** node.
2. In the content table, select the project.
3. In the **Properties** tab or window, double-click the **Packages** property value to display the Multi-Choice dialog window.
4. Check the **Standard Reports** checkbox and click **OK** to close the dialog window.

For each standard and custom report, the **Shared State** property value determines the report's visibility among the users in the project:

Private	The report is visible only to the creator, and only that user can modify the report. The report is not available to any other user. **Private** is the default value for each new report. **Private** also allows the report to be disabled temporarily, for example, during modifications.
Pending	The report is marked to be made visible to all users. The creator can set the **Pending** value. If appropriate, a project administrator can make the report visible by setting the **Public** value.
Public	The report is visible to all users. The **Public** value can be set by any user who has **Project Administrator** privilege for the project. A public report can be modified or deleted only by a project administrator.

Views

In the Systems Engineering and Requirements Management module, users can customize the display of property columns in the hierarchical content table. A unique set of columns, or *view*, can be applied to each project node or folder selected in the navigation tree. Views can be saved in the database and can be applied at any time.

All views are project specific and are stored in the **Reports and Formatting** folder. For each view, the **Shared State** property value determines the view's visibility among the users in the project:

Private — The view is visible only to the creator, and only that user can modify the view. The view is not available to any other user. **Private** is the default value for each new view.

Private also allows the view to be disabled temporarily, for example, during modifications.

Pending — The view is marked to be made visible to all users. The creator can set the **Pending** value. If appropriate, a project administrator can make the view visible by setting the **Public** value.

Public — The view is visible to all users. The **Public** value can be set by any user who has **Project Administrator** privilege for the project.

A public view can be modified or deleted only by a project administrator. If a deleted public view is the default view for one or more users, the **Default View** property for each user is automatically set to a blank value.

Private, public, and pending views can be exported from the Administration module as can other schema objects. However, only public views can be imported. For more information, see Exporting Project Data, Importing a Project, and Importing Schema Objects, later in this chapter.

Saved views are also available in the Basic pane of the Search module. For each search that you output to the Search Results dialog window, you can specify the column settings by selecting a view in the field at the bottom of the **Output Options** section. For more information about search results, see the *Systems Architect/Requirements Management User's Manual*.

Users and User Groups

Every person who uses Architect/Requirements must be represented by a *user* object in the database. A *maximum privilege level* assigned to each user limits that user's access to all projects. No user can be granted access to any project at a level that is higher than that user's maximum privilege level. The maximum privilege level also determines the type of Architect/Requirements license that is consumed for the user.

The project administrator can create *user groups* consisting of shortcuts to existing user objects in the project. Users represented by the shortcuts can be managed directly from the user group and changes are applied to the actual user objects. User groups can be added to a security profile to control user access for objects to which the security profile applies.

Security Profiles

A security profile is a set of access rules that control user actions on specific objects in a project. Every object can be associated with a security profile, and a given security profile can be applied to any number of objects.

Each security profile lists the users and user groups that have access to the objects to which the security profile applies. Within that list, individual users and user groups are assigned *permission* to view, modify, or delete the objects.

Activators

Activators are used to program automated processes that execute when certain events occur as a result of user actions in the Architect/Requirements client. Activators contain executable code written in Tool Command Language (Tcl). When an activator is triggered, that Tcl script is executed. For more information about using activators, see the *Systems Architect/Requirements Management API Reference* manual.

To open an activator script for editing or viewing, a user must have one of the following:

- The maximum privilege level of **Enterprise Administrator**.
- **Project Administrator** access privilege for the project that contains the activator.
- **Script Authoring** privilege for the project that contains the activator.

The same conditions must be met to view the **Script** property value of an activator. For more information, see Project Access and Special Privileges in chapter 5, *Managing Users*.

Administration Folders

In the Administration module, certain system defined folders are generated automatically for every new project. These system defined folders reside at the project's primary level, directly below the project node, and contain the schema objects for the project. The primary folders are **Activators**, **Property Definitions**, **Reports and Formatting**, **Security Profiles**, **Type Definitions**, and **Users**. For more information, see Schema Objects, earlier in this chapter.

The primary folders cannot be renamed, moved, or deleted. However, you can use subordinate folders to arrange schema objects according to your preference. Within any primary folder except the **Type Definitions** folder, you can do the following:

- Create subfolders, in which you can organize schema objects in hierarchies. For more information, see Creating a Subfolder, later in this chapter.

  You cannot create subfolders in the **Type Definitions** folder because the object types and subtypes are structured in a system defined hierarchy.

- Modify folders by doing the following:
  - Move existing schema objects from their primary folder to one or more subfolders. You can also move objects from one subfolder to another and from subfolders to the primary folder.
  - Move one or more subfolders within their containing primary folder.
  - Rename subfolders.

  For more information, see Modifying Administration Folders, later in this chapter.
- Control access to primary folders and subfolders through security profiles. For more information, see Applying a Security Profile to an Administration Folder, later in this chapter.
- Export and import project data from and to subfolders. For more information, see Exporting Project Data and Importing Schema Objects, later in this chapter.
- Delete subfolders. For more information, see Deleting Subfolders, later in this chapter.

Creating a Subfolder

You can create subfolders at the top level of any primary folder except the **Type Definitions** folder. Also, you can create subfolders in other subfolders to extend the hierarchy of schema objects to lower levels. You cannot create subfolders directly below the project node.

After you create a subfolder, you can rearrange existing schema objects by moving them to the new subfolder. You can also create new schema objects in the new subfolder, of the types appropriate to the primary folder. For more information, see Modifying Administration Folders, later in this chapter; Creating a Property Definition in chapter 3, *Customizing Object Types*; Creating a Document Template, Creating an Object Template, Creating a Style Sheet, Creating an Excel Template Object, and Creating a Stencil Object in chapter 4, *Customizing the Interface With Microsoft Office*; Creating a New User and Creating a User Group in chapter 5, *Managing Users*; and Creating a Security Profile in chapter 6, *Maintaining Project Security*.

All subfolders receive the **Admin Folder** object type. You cannot change that type, nor can you create subtypes for administration folders.

To create a subfolder:

1. Select the parent folder in the navigation tree or in the content table.

   Do not select the project in the navigation tree or a primary folder in the content table.

2. Do one of the following:

   - In the navigation tree, pull down the **File** menu and choose the **New→Folder** options.
   - In the content table:
     - To create a sibling of an existing subfolder within the parent folder, select the sibling, and then pull down the **File** menu and choose the **New→Folder** options.
     - To create a child of the parent folder, pull down the **File** menu and choose the **New→Child Folder** options.

   The content table displays the new folder with the default name in an open text field.

3. Enter the folder name in the text field, and then press the enter key.

   You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Procedure Notes

Step 1: To see lower level folders, click the plus signs, or double-click a folder with a plus sign. In the navigation tree, you can also select a folder and then pull down the **View** menu and choose **Expand**, or right-click the folder and choose **Expand** from the popup menu. In the content table, you can also select a folder and then pull down the **View** menu and choose **Expand All**, or right-click the folder and choose **Expand All** from the popup menu.

Step 2: You can also right-click the parent or sibling and choose the options from the popup menu. Or, click the **Create New Folder** or **Create New Child** button on the toolbar. In the navigation tree, you can also press control-L. In the content table, you can also press control-L for a new sibling or control-K for a new child.

Modifying Administration Folders

Administration folders allow you to create hierarchical structures for the schema objects in a project. In every primary folder except the **Type Definitions** folder, you can distribute schema objects among subfolders at multiple levels.

Schema objects can be moved from their primary folder to subfolders, from one subfolder to another, and from subfolders to the primary folder. In addition, subfolders can be moved and renamed.

- Schema objects and subfolders cannot be moved between primary folders or to the project's primary level, directly below the project node.
- Primary folders cannot be moved or renamed.
- Administration folders cannot be copied. Schema objects cannot be copied, except that user objects can be added to user groups through the copy function.

To move schema objects or subfolders:

1. Select the objects or subfolders, and then pull down the **Edit** menu and choose **Cut**.

   You can also use the mouse to drop the selection on the destination folder and complete this procedure.

2. Select the destination folder in the navigation tree or the content table, and then pull down the **Edit** menu and choose **Paste**.

   If the destination is a primary folder, you must select it in the navigation tree.

Procedure Notes

Step 1: In the navigation tree, you can select only one subfolder. In the content table, you can select multiple objects, including subfolders. You can also right-click the selection and choose **Cut** from the popup menu, click the **Cut** button on the toolbar, or press control-X.

Step 2: You can also right-click the selection and choose **Paste** from the popup menu. Or, click the **Paste** button on the toolbar or press control-V. To reverse this action, you can pull down the **Edit** menu and choose **Undo Move**, click the **Undo** button on the toolbar, or press control-Z.

To rename a subfolder:

1. Select the subfolder in the navigation tree or the content table, and then pull down the **File** menu and choose **Rename**.
2. Enter the new name in the open text field, and then press the enter key.

Procedure Notes

Step 1: You can also right-click the subfolder and choose **Rename** from the popup menu. Or, press the F2 key.

Step 2: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Applying a Security Profile to an Administration Folder

By default, users must have **Project Administrator** privilege for a project to modify the contents of an administration folder. However, all users in the project can view the contents of a folder.

Without defining additional project administrators, you can use security profiles to allow certain users to modify the objects in administration folders. You can also specify the users who can view the contents of a folder. For more information, see chapter 6, Maintaining Project Security.

You can apply a security profile to any administration folder, including a primary folder.

To apply a security profile to an administration folder:

1. Select the folder in the content table.

   The **Properties** tab or window displays all viewable properties for the folder.

2.

   In the **Value** column, double-click the **Security Profile** property value to display the Single-Choice dialog window.

3. Check the checkbox for the security profile, and then click **OK** to close the dialog window and set the property value.

   You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Deleting Subfolders

A subfolder cannot be deleted unless it is empty. First, you must move or delete all objects from each subfolder that you intend to delete. For more information, see Modifying Administration Folders, earlier in this chapter, and Deleting a Schema Object, later in this chapter.

To delete subfolders:

1. Select the subfolders.

   In the navigation tree, you can select only one subfolder. In the content table, you can select one subfolder or a group of nonadjacent or adjoining subfolders.

2. Pull down the **File** menu and choose **Delete**.

   A confirmation message asks if you are sure you want to delete the selected objects.

3. Click **Yes** to remove the selection from the Architect/Requirements database.

   This action clears the queue for the **Undo** option and cannot be reversed.

Procedure Notes

Step 2: You can also right-click the folder and choose **Delete** from the popup menu. Or, click the **Delete** button on the toolbar, or press the delete key.

Custom Menus

A system defined administration folder subtype, **Menu Folder**, lets you define project-specific menus and submenus. Using a system defined activator subtype, **Menu Item**, you can add options to main menus and submenus.

A given menu can be set for display in the Systems Engineering and Requirements Management module, the Search module, and the Administration module. Custom main menus are displayed between the **Tools** menu and the **Help** menu, in alphabetical order from left to right. Menu items and submenus are arranged alphabetically from top to bottom.

Custom menus and menu options cannot be internationalized.

Creating a Menu

Menu folders for main custom menus reside at the top level of the **Activators** folder. Submenus can be added by nesting other menu folders within a menu folder.

To create a menu:

1. Do one of the following:
   - For a main menu, select the **Activators** folder in the navigation tree.
   - For a submenu, select the parent menu folder in the navigation tree.
2. Pull down the **File** menu and choose the **New→Menu Folder** options.

   The content table displays the menu folder with the default name in an open text field.
3. Enter the menu folder name in the text field, and then press the enter key.

The displayed menu name is set by the **Menu Text** property of the menu folder object. If this value is blank, the name of the menu folder object is used for the menu. For more information, see Assigning a Display Name to a Menu or Menu Item, later in this chapter.

Procedure Notes

Step 2: You can also right-click the **Activators** folder or parent menu folder and choose the **New→Menu Folder** options from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Creating a Menu Item

You use a system defined activator subtype, **Menu Item**, to add individual options to a custom menu. For each menu item, you use Tool Command Language (Tcl) to write the script that executes the command. For information about editing activators, see the *Systems Architect/Requirements Management API Reference* manual.

To create menu items and edit their activator scripts, you must have **Script Authoring** privilege for the project. For more information, see Project Access, Special Privileges, and Licensing in chapter 5, Managing Users.

To create a menu item:

1. In the navigation tree or the content table, select the parent menu folder.
2. Pull down the **File** menu and choose the **New→Activator Subtype→Menu Item** options.

   The content table displays the item with the default name in an open text field.
3. Enter the menu item name in the text field, and then press the enter key.

The item name displayed on the menu is set by the **Menu Text** property of the menu item object. If this value is blank, the name of the menu item object is used for the item on the menu. For more information, see Assigning a Display Name to a Menu or Menu Item, later in this chapter.

Procedure Notes

Step 2: In the navigation tree, you can also right-click the parent menu folder and choose the **New→Activator Subtype→Menu Item** options from the popup menu. In the content table, you can also right-click the parent menu folder and choose the **New→Menu Item** options from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Assigning a Display Name to a Menu or Menu Item

For each menu folder object and menu item object, the **Menu Text** property value determines the text that is displayed as the name of the menu or item. If the **Menu Text** value is blank, the object name is used for the name of the menu or item. You can assign a display name that is different than the object name.

To assign a display name to a menu or menu item:

1. Within the **Activators** folder, select the menu folder or menu item.

   You can select a menu folder in the navigation tree or the content table. You can select a menu item only in the content table.
2. To display the Edit Properties dialog window, pull down the **File** menu and choose **Properties**.

   You can use the **Properties** tab or window if your selection is in the content table.
3. In the **Value** column, double-click the **Menu Text** value to open a text field.
4. Enter the text to display as the name, and then press the enter key.

Setting the Visibility of a Menu or Menu Item

A system defined property, **Shared State**, determines whether a menu or menu item is visible to the users in the project.

To set the visibility of a menu or menu item:

1. Within the **Activators** folder, select the menu folder or menu item.

   You can select a menu folder in the navigation tree or the content table. You can select a menu item only in the content table.

2. To display the Edit Properties dialog window, pull down the **File** menu and choose **Properties**.

   You can use the **Properties** tab or window if your selection is in the content table.

3.

   In the **Value** column, double-click the **Shared State** property value to display the Single-Choice dialog window.

4. Check one of the following checkboxes:

   Private — The menu or item is visible only to the creator. For any other user, the menu or item is not displayed. **Private** is the default value for each new menu and item.

   Private allows the user to disable the menu or item temporarily, for example, while completing the script that runs a menu item.

   Pending — The menu or item is marked to be made public by a project administrator. The creator can set the value to **Pending**.

   Public — The menu or item is visible to all users in the project. Any user who has **Project Administrator** privilege can change the value to **Public**.

   If a menu contains no public menu items, the entire menu is visible only to its creator. A menu is not displayed if it does not contain any menu items.

Filtering Menus by Privilege

A menu command may not work for a particular user if the user does not have the privilege for the object that the command accesses. For example, a user who does not have Architect privilege cannot use a command that modifies a building block. To avoid confusion, these commands can be removed from the user's custom menu structure. Setting the **Required Privilege** property on a menu folder restricts that menu to users that have the specified privileges. The choices in **Required Privilege** match the choices in the **Addition Privilege** property. When **Required Privilege** is set on a menu folder, that menu does not appear for users that do not have the corresponding privileges set in their user object's **Additional Privilege** property. If no privilege is set then all users have access to the menu.

The **Required Privilege** property only exists on menu folders. It is only possible to filter out an entire menu or sub-menu; individual commands cannot be filtered.

Filtering Menus with a Security Profile

It is possible to control which users see a particular menu or menu command. This is done by setting the **Security Profile** property on the corresponding menu folder or menu item. If a security profile is set, a user must have at least read access in the security profile, otherwise the menu or command does not appear. If no security profile is set, then all users have access to the menu or command. When filtering menus, security profiles are processed differently than when checking for other types of access. Project administrators have access to objects in a project even if the object has a security profile where they are not included. When filtering menus, you must include a user explicitly in the security profile else the user does not see the menu. This means menu filtering applies to administrators.

To grant all users access to a menu or command that has a security profile, set the **Everyone** choice in the security profile's **Read Access** property.

Setting the Module Display for a Menu

A given custom menu can be displayed in or removed from the Systems Engineering and Requirements Management module, the Search module, and the Administration module.

To set the module display for a menu:

1. Within the **Activators** folder, select the menu folder in the navigation tree or the content table.
2. To display the Edit Properties dialog window, pull down the **File** menu and choose **Properties**.

   You can use the **Properties** tab or window if your selection is in the content table.
3. In the **Value** column, double-click the **Module** property value to display the Multi-Choice dialog window.
4. Do one or both of the following:
   - Check the checkbox for each module where you want display the menu.
   - Clear the checkbox for each module where you want to remove the menu.

   You can check or clear the checkboxes in any combination.

Rearranging Menus and Items

Custom main menus are displayed between the **Tools** and **Help** menus, from left to right in alphabetical order of the top level menu folders in the **Activators** folder. Submenus and menu items are displayed in alphabetical order of object names within the menu folder hierarchy.

Any display name assigned to a menu folder object or a menu item object appears in alphabetical order of the object name. For more information, see Assigning a Display Name to a Menu or Menu Item, earlier in this chapter.

To rearrange a menu or menu item:

- Within the **Activators** folder, rename the object according to the new position.

Notebook Pane

As in the Systems Engineering and Requirements Management module, the notebook pane displays information for the object selected in the navigation tree or the content table. In the Administration module, the notebook pane contains the **Properties**, **Attachments**, **Preview**, and **Where Used** tabs. Each tab can be opened in a separate floating window.

Properties Tab

The **Properties** tab displays all viewable properties that apply to the selected schema object. Values that you cannot change are dimmed in the **Value** column. To change an editable property value, you can double-click the value to open the Single-Choice dialog window, the Multi-Choice dialog window, or a text field, depending on the property type.

The **Properties** tab has controls that filter the properties displayed in the table. You can use these controls to view properties according to certain categories. For more information, see Filtering the Properties Tab, later in this chapter, and chapter 2, Customizing Object Properties.

Attachments Tab

The **Attachments** tab displays the notes attached to the selected schema object. You can attach notes to schema objects in the same way as you do to objects in the Systems Engineering and Requirements Management module. You can also attach notes to other notes.

You cannot attach notes to individual user objects. However, you can attach notes to user groups.

When you create a note, you choose one of the following content formats:

- Plain text, shown by the **Text** value of the note's **Text Format** property.
- Rich text, shown by the **HTML** value of the note's **Text Format** property.
- Microsoft Word, shown by the **OpenXML** value of the note's **Text Format** property.

You can also assign a user defined subtype to the note. For more information, see Creating a Subtype in chapter 3, *Customizing Object Types*.

This tab displays the notes hierarchically in the **Attachment** column. This column contains a plus sign (+) for each note to which at least one other note is attached. You can click the plus sign to see the notes at the next lower level. Plus signs may be displayed also for lower level notes.

For more information about working with notes, see the *Systems Architect/Requirements Management User's Manual*.

Preview Tab

The **Preview** tab displays the content of the style sheet, template, activator, or macro selected in the content table. Also, you can open the **Preview** floating window to view the content of the note selected in the **Attachments** tab or window. For more information, see Floating Tab Windows, later in this chapter.

Where Used Tab

The **Where Used** tab displays all objects that reference the selected schema object. Referencing objects shown in the tab depend on the selected object type:

Object Type	Referencing Objects
Activator	Type definitions
Property definition	Type definitions
Document template	Folder instances, saved searches
Object template	Type definitions, document templates
Style sheet	Document templates
View	Folder type definitions, folder instances, user objects
User	User groups, security profiles, projects
User group	Security profiles
Security profile	Objects to which the security profile applies

If a referencing object resides in the Systems Engineering and Requirements Management module, you can navigate to that object by selecting it in the **Where Used** tab, pulling down the **View** menu, and choosing the **Go To→Go To Object** options. Or, right-click the object in the tab and choose **Go To Object** from the popup menu. You cannot automatically navigate to referencing objects in the Administration module.

Floating Tab Windows

You can open the top tab in a separate floating window by clicking the **Open tab** button on the **Notebook** pane toolbar. To open all tab windows simultaneously, click the **Open each tab** button. Also on the toolbar, you can:

- Click the **Close all** button to close each open tab window in a single action.
- Click the **Minimize** button to reduce the **Notebook** pane to its minimum height.
- Click the **Maximize** button to enlarge the **Notebook** pane to its maximum height.

You can see a description of each button by resting the pointer on the button to display a tooltip.

Filtering the Properties Tab

In the **Properties** tab and floating window, you can set the properties that you want to see in the property table. By default, the table displays all viewable properties for the selected object or objects.

You can limit the properties to a specific subset through filtering options on the **View** and popup menus and through filter buttons in each view. For more information, see appendix System-Defined Properties in the Administration Module, and Overview of Property Definitions in chapter **Unsatisfied xref reference**, ***Unsatisfied xref title***.

The filter buttons in the **Properties** tab are hidden each time you start a new Architect/Requirements session. In the floating window, the buttons are hidden each time you open the window.

To show the filter buttons:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→Show Controls**.
- Right-click the table and choose **Filter→Show Controls** from the popup menu.
- Click the down arrow on the horizontal split bar above the table.

You can also point to the split bar until the pointer becomes a double vertical arrow. Then, click the split bar to size the filter bar automatically, or drag the split bar to set the size that you want.

To hide the filter buttons:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→Hide Controls**.
- Right-click the table and choose **Filter→Hide Controls** from the popup menu.
- Click the up arrow on the horizontal split bar above the table.

To display only properties whose values you can change:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→Editable**.
- Right-click the table and choose **Filter→Editable** from the popup menu.
- Click the **Editable** filter button.

To display only properties whose values cannot be changed:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→Read-only**.
- Right-click the table and choose **Filter→Read-only** from the popup menu.
- Click the **Read Only** filter button.

To display only the custom properties for the project:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→User Defined**.
- Right-click the table and choose **Filter→User Defined** from the popup menu.
- Click the **User Defined** filter button.

To display only system defined properties:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→System**.
- Right-click the table and choose **Filter→System** from the popup menu.
- Click the **System** filter button.

To display all viewable properties:

Do one of the following:

- Pull down the **View** menu and choose **Filter Properties→All**.
- Right-click the table and choose **Filter→All** from the popup menu.
- Click the **All** filter button.

Creating a Project

Only enterprise administrators can create projects. By default, the creator becomes the original project administrator, with full access to perform any action in the project and to delete the project. In addition, all other enterprise administrators are added to the **Users** folder in the new project and have full access automatically.

You must have **Enterprise Administrator** privilege to perform this procedure.

For any other user, access must be granted specifically after the project's creation. Access can be granted by the creator, by another enterprise administrator, or by a user who is subsequently granted **Project Administrator** access privilege. For more information, see Project Access and Granting or Revoking Project Access in chapter 5, *Managing Users*.

- To automatically create a full project with objects in both the Systems Engineering and Requirements Management and Administration modules, you can import object data from an XML file. For more information, see Importing a Project, later in this chapter.
- The new project can be deleted by users who are subsequently granted **Project Administrator** access privilege. For more information, see Deleting a Project, later in this chapter, and Modifying a User Profile in chapter 5, *Managing Users*.

To create a project:

1. In the navigation tree, select the root node.

   The content table displays all projects in the database.

2. Pull down the **File** menu and choose the **New→Project** options.

   The content table displays the project with a default name in an open text field.

3. Enter the project name, and then press the enter key.

   To refresh the view, pull down the **View** menu and choose **Refresh**. You can also right-click the root node and choose **Refresh** from the popup menu. Or, click the **Refresh** button on the toolbar.

Procedure Notes

Step 2: You can also right-click the root node and choose the **New→Project** options from the popup menu. Or, right-click in the content table and choose **New Project** from the popup menu.

Importing a Project

An enterprise administrator can create an entire project in one action by importing data from an XML file. The new project automatically contains objects in both the Systems Engineering and Requirements Management module and the Administration module.

You must have **Enterprise Administrator** privilege to perform this procedure.

The import file can be one containing data that was previously exported from the Administration module. For more information, see Exporting Project Data, later in this chapter. Or, you can use the data in an XML file exported from a product other than Architect/Requirements.

The file being imported must have been exported from either the same Architect/Requirements version or the prior major version. Exporting from a newer version and importing into an older version is not supported because it may not be possible to map new data into an older database.

For the Systems Engineering and Requirements Management module, the data represents folders, requirements, building blocks, groups, notes, and diagrams, complete with the following:

- Properties and values, both system defined and user defined.
- Parent, child, and sibling relationships among imported objects.
- Trace links between objects that reside within the project from which the data was exported. Interproject trace links are not imported.
- Versions and variants of requirements and building blocks.
- **Create User** and **Create Time** system defined properties of the objects. If the created user does not exist in the database, then the current user is set as the created user. For more information, see the Table of System-Defined Properties in the appendix System-Defined Properties in the Administration Module.

In the export file, a diagram may contain shapes representing objects that reside outside the diagram's parent folder. These cross-folder or cross-project references are not preserved in the imported diagram, and the corresponding shapes are marked in red.

User groups cannot be imported into projects.

For the Administration module, the file contains data that defines the following schema objects:

- Activators and macros in the **Activators** folder.
- Property definitions in the **Property Definitions** folder.
- Document templates, object templates, style sheets, and reports in the **Reports and Formatting** folder.
- Security profiles in the **Security Profiles** folder.
- Object types and subtypes in the **Type Definitions** folder.

Data is not imported for user objects, proxy objects, or links that are external to the project, such as Teamcenter Interface links.

Schema objects defined in the file are created if they do not exist in the database. Existing schema objects are updated in the database as defined in the file.

- Before importing, ensure that you know the name and location of the XML file containing the data that you want.
- Although the import file can be opened in a text editor such as Microsoft WordPad or Notepad, changing the data is an advanced operation. Siemens PLM Software recommends that you do not open the import file.

To import a project:

1. Select the root node in the navigation tree, and then pull down the **File** menu and choose the **Import→Project** options.

   Architect/Requirements displays the Open dialog window, which lists existing folders and files in the current drive or folder.

2. Select the import file in the list, or enter the name in the **File name** field.

   The **Look in** field displays the current drive or folder. If the import file is not in the list, you can use this field to change the drive or folder.

3. Click **Open** or press the enter key.

   This action clears the queue for the **Undo** option and cannot be reversed.

   The dialog window closes, and a message indicates that the import is in progress. When the import is complete, a confirmation message is displayed, and the new project node is displayed in the navigation tree and the content table.

4. Select the project node, and then pull down the **File** menu and choose **Rename**.
5. Enter the project name in the open text field, and then press the enter key.

Procedure Notes

Step 1: You can also right-click the root and choose the options from the popup menu.

Step 4: You can also right-click the project and choose **Rename** from the popup menu. Or, press the F2 key. To reverse this action, pull down the **Edit** menu and choose **Undo Rename**.

If you are importing one or more projects from Architect/Requirements versions prior to 8.2, perform the upgrade steps mentioned in the topic *Converting Excel Templates* described in Chapter 3 of the *Architect/Requirements Management Server Installation Manual*. You can also use the bulk conversion method for converting the templates. The bulk conversion method ignores templates from existing projects that are already in the Open XML format.

Exporting Project Data

Data representing objects in a project can be exported from the Administration module to an XML file. The export file is generated by Architect/Requirements and is saved in the drive or folder that you specify. Then the data can be imported to another project, where Architect/Requirements reproduces the objects. Thus, objects can be shared among projects in the same Architect/Requirements installation and in installations at other sites. For more information, see Importing a Project and Importing Schema Objects, later in this chapter.

Depending on your purpose, you can:

- Export all data for the entire project. From the Systems Engineering and Requirements Management module, the data represents all of the project's folders, requirements, building blocks, groups, notes, and diagrams. The object data is complete with the following:
    - Properties and values, both system defined and user defined.
    - Parent, child, and sibling relationships among the objects.
    - Trace links between objects that reside within the project. Interproject trace links are not exported.
    - Versions and variants of requirements and building blocks.

    A diagram may contain shapes representing objects that reside outside the diagram's parent folder. Although these cross-folder or cross-project references are exported to the XML file, they are not preserved when the diagram is imported.

    In addition, the file includes all of the project's schema data, which defines the following schema objects in the Administration module:

    - Activators and macros in the **Activators** folder.
    - Property definitions in the **Property Definitions** folder.
    - Document templates, object templates, style sheets, and reports in the **Reports and Formatting** folder.
    - Security profiles in the **Security Profiles** folder.
    - Object types and subtypes in the **Type Definitions** folder.

    A project administrator can import this file to create a new project automatically, complete with predefined objects in both modules.

- Export schema data only, representing schema objects in the Administration module. To a given file, you can export data for either of the following:
    - All schema objects in the project, including both system defined and user defined schema objects.
    - Selected schema objects within a folder, where you can select one object or a group of nonadjacent or adjoining objects.

    A project administrator can import this file to initialize a new project with predefined schema, or to update the schema of an existing project.

- Data is not exported for user objects, proxy objects, or links that are external to the project, such as Teamcenter Interface links.

- The **.xml** file name extension is assigned to all files containing data exported from the Administration module. Siemens PLM Software recommends that you give each export file a name that clearly identifies the data.

To export project data:

1.

   Do one of the following:

   - To export the entire project, select the project node in the navigation tree, and then pull down the **File** menu and choose the **Export→Project** options.
   - To export all schema data in the project, select the project node in the navigation tree, and then pull down the **File** menu and choose the **Export→Schema** options.
   - To export selected schema data:

     . Open the containing folder, and then select the schema objects in the content table.

     You can select both nonadjacent and adjoining objects.

     . Pull down the **File** menu and choose the **Export→Schema** options.

   Architect/Requirements displays the Save dialog window, which lists existing folders and files in the current drive or folder.

2. In the **File name** field, enter a name that clearly identifies the exported data.

   The **Save in** field displays the current drive or folder. You can save the file in the displayed location, or you can use this field to change the drive or folder.

3. Click **Save** or press the enter key.

   The dialog window closes, and a message is displayed to indicate that the export is in progress. When the export is complete, the message closes, and the file is saved in the specified location.

   Although the export file can be opened in a text editor such as Microsoft WordPad or Notepad, changing the data is an advanced operation. Siemens PLM Software recommends that you do not open the export file.

Procedure Notes

Step 1: You can also right-click the project node in the navigation tree and choose the options from the popup menu. For selected schema data, you can also right-click the selection in the content table and choose **Export Schema** from the popup menu.

Step 2: The **.xml** file name extension is added automatically. You can also select an existing XML file in the list to overwrite that file.

Importing Schema Objects

Using data exported from the Administration module to an XML file, a project administrator can easily reproduce one project's schema objects in another project. Objects defined in the file are created if they do not exist in the database. Existing objects are updated in the database as defined in the file.

The file being imported must have been exported from either the same Architect/Requirements version or the prior major version. Exporting from a newer version and importing into an older version is not supported because it may not be possible to map new data into an older database.

The following schema objects can be imported to the Administration module:

- Activators and macros in the **Activators** folder.
- Property definitions in the **Property Definitions** folder.
- Document templates, object templates, style sheets, and reports in the **Reports and Formatting** folder.
- Security profiles in the **Security Profiles** folder.
- Object types and subtypes in the **Type Definitions** folder.

Data is not imported for proxy objects, user objects, or links that are external to the project, such as Teamcenter Interface links.

- Before importing, ensure that you know the name and location of the XML file containing the data that you want.
- Although the import file can be opened in a text editor such as Microsoft WordPad or Notepad, changing the data is an advanced operation. Siemens PLM Software recommends that you do not open the import file.

To import schema objects:

1. Select the project node in the navigation tree, and then pull down the **File** menu and choose the **Import→Schema** options.

   Architect/Requirements displays the Open dialog window, which lists existing folders and files in the current drive or folder.

2. Select the import file in the list, or enter the name in the **File name** field.

   The **Look in** field displays the current drive or folder. If the import file is not in the list, you can use this field to change the drive or folder.

3. Click **Open** or press the enter key.

   This action clears the queue for the **Undo** option and cannot be reversed.

   The dialog window closes, and a message indicates that the import is in progress. When the import is complete, a confirmation message is displayed.

Procedure Notes

Step 1: You can also right-click the project and choose the options from the popup menu.

Running an Import From the Command Line

You can enter the **tcradmin** script on the command line to import data to the Architect/Requirements database.

The following example shows the **tcradmin** syntax:

```
tcradmin -action import -type <typeOfImport> -file <filename> -loid <importOwner> -
user <username> -password <password> [-key <installationKey>] [-location
<originalFileName>] [-subtype <subtype>]
```

Table 1-1 describes the **tcradmin** arguments.

Table 1-1. tcradmin Arguments for Import From the Command Line

Argument	Description	
–type	Import action to perform:	
	XML	Import a folder of data in XML format.
	XML_UPDATE	Import and merge a folder of data in XML format.
	PROJECT	Import an entire project in XML format.
	SCHEMA	Import a project's schema or selected schema objects.
	MS_WORD	Import a Word document. This is the default.
	STYLESHEET	Import a Word document into a style sheet object.
	MS_EXCEL	Import an Excel spreadsheet.
	EXCEL_TEMPLATE	Import an Excel template.
	AP233	Import data in AP233 format.
-file	Complete path of the import file.	
-loid	LOID of the object to import into.	
–user	Architect/Requirements user name under which you want to log in to the server.	
	To import an entire project, this user name must have **Enterprise Administrator** privilege.	
-password	Architect/Requirements password for the user name under which you want to log in to the server.	
–key	Web server installation identifier used to look up Web application parameters such as **ImportExportDir**.	

Table 1-1. tcradmin Arguments for Import From the Command Line

Argument	Description
	 This argument defaults to *machinename* + **"1"**, which is correct in most circumstances. This argument may be needed if there are no or more than one Architect/Requirements Web servers installed on this machine.
`–location`	Original Word document file name to store as property on imported requirements.
`–subtype`	Requirement subtype to use for requirements imported from Word.
`-logToStdOut`	Redirects the output of the **tcradmin** command to **stdout**. By default, most of the output of **tcradmin** is written to the Architect/Requirements server log file, **TcrServerLog.html**. The Architect/Requirements server log is written in a tabular HTML format that is convenient to view. When redirected to **stdout**, the Architect/Requirements log information is written as plain text.

Running an Export From the Command Line

You can enter the **tcradmin** script on the command line to export data from the Architect/Requirements database.

The data is exported to a temporary file with a system-generated name. The last line of script output displays the path to the export file.

The export file location is specified by the **ImportExportDir** configuration parameter. For more information about the **ImportExportDir** parameter and customizing the Architect/Requirements server, see the *Systems Architect/Requirements Management System Administrator's Manual*.

The following example shows the **tcradmin** syntax:

```
tcradmin -action export –type <typeOfExport> -loid <objectsToExport> -user
<username> -password <password> [-template <outputTemplate>] [-props <properties>]
[-live] [–notDeep] [–noOLE] [-key <installationKey>]
```

Table 1-2 describes the **tcradmin** arguments.

Table 1-2. tcradmin Arguments for Export From the Command Line

Argument	Description	
`–type`	Export action to perform:	
	`XML`	Export data in XML format.
	`PROJECT`	Export an entire project in XML format.
	`SCHEMA`	Export a project's schema or selected schema objects.
	`MS_WORD`	Export a Word document. This is the default.
	`MS_EXCEL`	Export an Excel spreadsheet.
	`AP233`	Export data in AP233 format.
	`TC_XML`	Export data for migration to Teamcenter.
`-loid`	LOID or comma-separated list of LOIDs of the objects to export.	
`–user`	Architect/Requirements user name under which you want to log in to the server.	
`-password`	Architect/Requirements password for the user name under which you want to log in to the server.	
`–template`	The document or Excel template to use. Valid only for the MS_WORD and MS_EXCEL export types.	
`-props`	Properties to include as column headings in a Microsoft Office Excel export file.	

Table 1-2. tcradmin Arguments for Export From the Command Line

Argument	Description
`–live`	Make the exported Excel spreadsheet live.
`–notDeep`	Export only the specified objects to Microsoft Office Word and do not include their descendents.
`–noOLE`	Do not include OLE objects in a Microsoft Office Word export file.
`–key`	Web server installation identifier used to look up Web application parameters such as **ImportExportDir**.  This argument defaults to *`machinename`* `+ "1"`, which is correct in most circumstances. This argument may be needed if there are no or more than one Architect/Requirements Web servers installed on this machine.
`-logToStdOut`	Redirects the output of the **tcradmin** command to **stdout**. By default, most of the output of **tcradmin** is written to the Architect/Requirements server log file, **TcrServerLog.html**. The Architect/Requirements server log is written in a tabular HTML format that is convenient to view. When redirected to **stdout**, the Architect/Requirements log information is written as plain text.

Managing Rule File to Export or Import Objects

Architect/Requirements provides mechanisms to export and import the Architect/Requirements objects and their properties in XML format that is compliant with the AP233 Part 28 standard. The Part-28 format of the AP233 standard provides many constructs (collection of XML elements) that can be used to represent the Architect/Requirements object types and the property types in the XML format.

A Rule File:

- Is an XML file that describes the mapping from the Architect/Requirements object types and property types to the AP233 tags.
- Is a collection of rules for Architect/Requirements object types and property types of objects.

  Each rule is a collection of XML elements that the Architect/Requirements export operation processes to write objects to an XML file in the AP233 schema and the Architect/Requirements import operation processes to create or update the Architect/Requirements objects from an AP233 XML file.
- Can be used to customize the behavior of the export and import operations.

Using the Rule File

Rule file is a collection of rules for objects. A rule contains AP233 schema tags and some XML tags (such as the **<Method>** tag) that are used by Architect/Requirements for processing.

The **Default AP233 Rule File** is located in the **Reports and Formatting** folder of every project in the Administration module. There is only one rule file for each project, to maintain data integrity and to reduce potential errors during export or import.

Project administrators can export a rule file by right-clicking he rule file and selecting the **Export Schema** or **Copy to client** option. They can import a rule file by right-clicking the rule file to be updated and selecting the **Import→AP233 Rule File** option.

A rule file can be used by the project administrators to:

- Specify rules for object types and their property types that are used during the export and import operations.
- Specify the properties of objects to be included or excluded during export to or import from an AP233 XML file.
- Specify the unique property for updating objects during import from an AP233 XML file.

Identifying the Rule for Objects

The name of the rule is either object type name or its subtype name. If a rule with a subtype name is not present, then the rule of its parent type is processed. For example, the rule for a requirement is processed if there is no rule for a paragraph.

An extract of the rule for building block objects is shown below:

```
<BuildingBlock>
<ap233:System>
<ExportRule>...</ExportRule>
<ImportRule>...</ImportRule>
</ap233:System>
...
<ap233:System_version>
<ExportRule>...</ExportRule>
<ImportRule>...</ImportRule>
</ap233:System_version>
...
<ap233:View_definition_context>...</ap233:View_definition_context>
...
<ap233:System_view_definition>...</ap233:System_view_definition>
...
</BuildingBlock>
```

The export process looks up the mapping specified in the rule file to determine what objects to export to an output file and how to export them. The import process looks up the mapping specified in the rule file to determine what objects to create or which properties to update from an input file.

In the preceding example, **<ap233:System>**, **<ap233:System_version>**, **<ap233:View_definition_context>**, and **<ap233:System_view_definition>** are the AP233 elements that are used to describe building blocks. Similarly, other Architect/Requirements object types are described using such AP233 elements in the rule file.

The export rule for the building blocks in the selected project corresponds to the **<ExportRule>...</ExportRule>** element. Each **<ExportRule>** element contains AP233 and **<Method>** elements used to get data from Architect/Requirements. Similarly, the **<ImportRule>...</ImportRule>** element corresponds to the import rule for the building blocks. Each **<ImportRule>** element contains **<Method>** elements used to create or update data in Architect/Requirements.

Identifying the Rule for Properties of Objects

An object can have system defined properties and user defined properties. Similar to the rules for Architect/Requirements object types, there are separate rules for system defined and user defined properties of objects in Architect/Requirements. Each of these rules contains AP233 and other XML elements that describe the properties and their behavior during export and import operations.

A rule file contains rules for the following property types:

Property Type	**System defined**	**User defined**
Text	TextDefinition	TextProperty
Choice	ChoiceDefinition	ChoiceProperty
Date	DateDefinition	DateProperty
Numeric	NumericDefinition	NumericProperty

In a rule file, the name of the rule for properties is based on the property type. For example, the rule for a user defined text property is **TextProperty** and the rule for a system defined date property is **DateDefinition**.

Including and Excluding Properties

By default, all user defined properties are exported and imported. A rule element has **include** and **exclude** attributes that can be used to specify the properties of objects to be included or excluded during export to and import from an AP233 XML file.

To include any property for export and import, add the name of the property to the **include** attribute. Similarly, if you want to exclude a property, add it in the **exclude** attribute. The **include** attribute also supports the ***** wild card, which includes all matching properties.

For example, to include all the values of the **ChoiceProperty** property, except the **Baseline** property:

```
<ChoiceProperty include="*" exclude="Baseline">
```

To include all the values of **NumericProperty** property:

```
<NumericProperty include="*">
```

To include only the **Name** values in the **TextDefinition** property:

```
<TextDefinition include="Name">
```

To include only the **ROIN** and **Number** values in the **TextDefinition** property:

```
<TextDefinition include="ROIN,Number">
```

The above rule is used only during export and not during the import operation.

Updating Objects During Import

When you import from an AP233 XML file, the import process updates an object if it already exists in Architect/Requirements. Otherwise, a new object is created.

The import process determines whether to update an existing object or to create a new object, based on the following conditions:

- The import process checks whether the object being imported is present in Architect/Requirements within the selected folder or project.
- The property used to determine the uniqueness of the object is specified in the rule file.
- If there is an object present with the same value for the property specified in the rule file, then the import process does not create a new object.

For example, the following condition is used to evaluate the uniqueness of requirement objects:

```
<Condition name="isRequirementUnique">
           <Method name="getMatchingElement" static="false"
contextObject="importerObject" output="preserve">
                 <args>
                       <arg type="String" default="ROIN" />
                       <arg type="Vector" reverseRef="Of_product/
ap233:Requirement_version/Defined_version/
ap233:Requirement_view_definition/Described_element/
ap233:Assigned_property/Definition/ap233:Property_representation" />
                       <arg type="String"
default="Definition/ap233:Assigned_property/Name" />
                 </args>
           </Method>
           <Method name="findObject" static="false" contextObject="importerObject">
                 <args>
                       <arg type="Object" default="previousObject" />
                       <arg type="String"
default="Definition/ap233:Assigned_property/Name" />
                       <arg type="String" default="Rep/ap233:Representation/Items/
ap233:String_representation_item/String_value" />
                       <arg type="String" ref="Id" />
                       <arg type="String" default="text" />
                 </args>
           </Method>
   </Condition>
```

In this example, the condition for the requirement object is **isRequirementUnique** specified in the **<Condition name="isRequirementUnique">** element. Similarly, the condition for notes is **isNoteUnique** and the condition for building blocks is **is BuildingBlockUnique**. For trace links, there is no condition to be specified. The uniqueness of a trace link is determined by the end object to which it is attached.

Specifying the Unique Property for Updating Objects During Import

To change the property that is used to determine the uniqueness of an object, users should change the property name present in the first argument value in the **getMatchingElement** method. In the above example, the **<arg type="String" default="ROIN" />** element specifies **ROIN** as the property to check for the uniqueness of a requirement object to be imported. To illustrate, if Architect/Requirements contains a requirement object with the ROIN value 123 and the AP233 XML file being imported also has a requirement with the same ROIN value, then the import process does not create a new object. Also, the import process updates the properties of the existing object with the properties of the object in the imported file.

If the unique property is changed to **Name** instead of **ROIN**, then the import process looks for the name of the requirement to determine the uniqueness of the requirement object being imported.

If the new property to be used in the condition is of a different type, such as a user defined numeric property instead of ROIN (a text property), then the arguments in the **findObject** method should also be modified, in addition to the property name. An example of the arguments in the **findObject** method is shown below.

```
 <Method name="findObject" static="false" contextObject="importerObject">
                <args>
                       <arg type="Object" default="previousObject" />
                       <arg type="String"
default="Definition/ap233:Assigned_property/Name" />
                       <arg type="String" default="Rep/ap233:Representation/Items/
ap233:String_representation_item/String_value" />
                       <arg type="String" ref="Id" />
                       <arg type="String" default="text" />
                </args>
           </Method>
```

For a text property, the third argument should be changed to:

```
<arg type="String" default=”Rep/ap233:Representation/Items/
ap233:String_representation_item/String_value” />
```

For a numeric property, the value of the third argument should be:

```
<arg type="String" default=”Rep/ap233:Representation/Items/
ap233:Numerical_item_with_unit/Value_component/Any_number_value-wrapper” />
```

In addition, the fifth argument should be changed if text property is not specified in the third argument.

For a numeric property, the fifth argument should be changed to:

```
<arg type="String" default="number" />
```

For a choice property, the fifth argument should be changed to:

```
<arg type="String" default="choice" />
```

The date property is not supported for use as the property for checking the uniqueness of the object being imported.

Architect/Requirements XML Format

This section describes the basic organization and content for the Architect/Requirements XML format that can be used for import or export operations. In many cases, the questions not addressed here can be answered by using the **Export→XML** command on an object of interest and then inspecting the file.

Types of XML Files

Architect/Requirements supports three types of XML files: schema, data, and project. XML files are read and written using the **importDocument** and **exportDocument** API methods. The type of the XML file determines the file type keyword to use. For more information, see the *Systems Architect/Requirements Management API Reference*.

- Schema XML file contains the Architect/Requirements objects from a project that are visible in the administration module. Schema XML files may contain all or only some of the administration objects. Use the **SCHEMA** keyword to import or export a schema.
- Data XML file contains the Architect/Requirements objects from a project that are visible in the requirements module. Data XML files may contain all or only some of the non-administration objects. Use the **XML** keyword to export data objects. Use the **XML** or the **XML_UPDATE**

keyword to import data objects. The **XML_UPDATE** keyword is available only through the API method and allows merging changes into existing objects.

- Project XML file contains the complete schema and data XML files for a project concatenated together. The files are separated by the ###### **TCR** ###### string. Because a project file actually contains two separate XML files, it is not a well-formed XML file. This means the project file may not be usable in some XML viewing utilities. In that event, the file can be manually split into two pieces (using Notepad or a similar text editor) for the XML viewer. Use the **PROJECT** keyword to import or export the entire project.

Types of Data Elements

The data file contains a single **<data>** element. The data element contains a flat list of project object elements. The tag for an object uses the database object type name (**DataBean.TYPE**), for example, **<RequirementDB>**. The data for an object is placed in the sub elements.

The three types of data elements are required fields, property values, and object type-specific fields.

- **Required Fields**

  These elements must be included for each object. If they are not included, the object is not imported. Even if a particular field has no value, its element must be included.

  - **name** is the name of the object. Note that XML special characters must be encoded. For example, > is specified as **>**.
  - **id** is the unique object identifier, used for referencing other objects, as with the **<owner>** tag. To update objects with the **XML_UPDATE** keyword, the **id** field must be the LOID of the existing object. For other imports, any value can be used, as long as it is unique within the XML file.
  - **owner** is the ID of the object's owner. For objects that do not have an owner, such as trace links, use **null**.
  - **typedefinition** is the name of the object's type definition (its subtype value). If there is no type definition with this name when an object is imported, it uses the base type instead.

  For example:

```
<name>R1</name>
<id>89.0.3649333</id>
<owner>89.0.3650243</owner>
<typedefinition>Requirement</typedefinition>
```

- **Property Values**

  Most property values, both system defined and user defined, are included using a property element. Attributes are used for the property name and the type (system or user). The element's value is the property value.

  For numeric and date properties, a **rawdata** attribute may also be used. Raw data is a normalized form of the value used to avoid localization and time zone issues when parsing. For dates, raw data is the timestamp. For numbers, raw data is the number in English locale.

  Read-only properties are included in exports; however, they are ignored on import. All the properties are optional for import. If a property is not included, it assumes its default value for

objects that are created, or retains its existing value for updates. Properties that are not initialized may have a blank or **null** value.

For example:

```
<property name="ROIN" type="system">0005</property>
<property name="Create Time" type="system" rawdata="1224344701202">
10/18/2008 10:45 AM</property>
<property name="Color" type="user">Blue</property>
```

- **Type-Specific Fields**

  Some object types support additional fields for type-specific data.

  Object types that can be versioned, such as requirements and building blocks, have the following fields:

  - **master** is the ID of the current version of the object.
  - **versionnumber** is the version number of the object. **0** is used for objects that have never been frozen.

  For example:

  ```
  <master>89.0.3649333</master>
  <versionnumber>0</versionnumber>
  ```

  Objects with rich text, such as requirements and notes, have the following fields:

  - **HTML** is the encoded text content.
  - **filename** is the internally used unique file name for referencing graphics and OLE objects.

  For example:

  ```
  <HTML></HTML>
  <filename>>/Requirement-0001-11d108331fd.html</filename>
  ```

  Linking objects, such as trace links and connections, have the fields:

  - **frontend** is the ID of the object that the link points to (complying end).
  - **backend** is the ID of the object that the link points from (defining end).
  - **linktype** is the type of the link (**DataBean.LINK**).

  For example:

  ```
  <frontend>89.0.3649333</frontend>
  <backend>89.0.3649284</backend>
  <linktype>Complying</linktype>
  ```

  Other type-specific fields can be obtained by examining the XML output for objects of that type.

- **Deleting Objects**

  When using the **XML_UPDATE** keyword, the existing objects can be deleted from the database by including `action="delete"` in the main object tag.

For example:

```
<BuildingBlockDB action="delete">
<id>410.0.11023</id>
<owner>410.0.10867</owner>
<typedefinition>Building Block</typedefinition>
</BuildingBlockDB>
```

Schema XML Format

The schema file contains a single **<schema>** element. The schema element contains a flat list of administration object elements. The tag for a schema object uses the database object type name (**DataBean.TYPE**), for example, **<UserTypeDefinitionDB>**.

Schema imports always attempt to merge with existing schema objects. Schema object references generally are by name instead of ID (the owner field is an exception). If an element in the XML file has the same name as an existing schema object of the same type, it updates that object. Where possible, the updates are non-destructive, that is, the data is added but not removed. For example, the import associates new properties with an existing type definition. However, the import does not remove the properties that are not included in the XML file.

Order of Objects

Forward object references in the XML files are not supported. Referencing another object can be done only if the referenced object appears earlier in the XML file. For example, a member of a folder must appear after the folder, and a trace link must appear after the two objects that are linked.

Removing References to a Schema Object

The **Where Used** tab displays the objects that refer to the object selected in the content table. For example, when you select a property definition, the tab shows the type definitions to which the property definition is applied. For more information, see Notebook Pane, earlier in this chapter.

To remove references to a schema object:

1. Select the object and click the **Where Used** tab to display the referencing objects.

   To see the locations of the objects, you can add the **Full Name** property in the tab. Right-click any column heading to display the Column Settings dialog window, check the **Full Name** checkbox, and click **OK**.

2. Depending on the schema object type and the referencing object type, do one of the following actions:

Schema Object	Referencing Object	Action
Activator	Type definition	a. In the **Type Definitions** folder, select the referencing type definition.

Schema Object	Referencing Object	Action
		b. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. c. Double-click the **Activators** value to display the Multi-Choice dialog window. d. Clear the checkbox for the activator that you want to remove, and then click **OK**.
Property definition	Type definition	e. In the **Type Definitions** folder, select the referencing type definition. f. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. g. Double-click the **Properties** value to display the Multi-Choice dialog window. h. Clear the checkbox for the property definition that you want to remove, and then click **OK**.
		 For certain property definitions, references have generic names that are not found in the **Type Definitions** folder. For **Shared State**, for example, references named **Menu Item** and **View** have no explicit type definitions. Such references are system-generated and cannot be removed.
Document template	Search	i. In the **Reports and Formatting** folder, select the referencing search. j. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. k. Double-click the **Document Template** value to display the Single-Choice dialog window. l. Clear the checkbox for the document template and click **OK**.
Document template	Folder instance	m. In the **Where Used** tab, select the referencing folder, and then pull down the **View** menu and choose the **Go To→Go To Object** options. The folder is selected automatically in the Systems Engineering and Requirements Management module.

Schema Object	Referencing Object	Action
		n. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. o. Double-click the **Document Template** value to display the Single-Choice dialog window. p. Clear the checkbox for the document template and click **OK**.
Object template	Document template	q. In the **Reports and Formatting** folder, select the referencing document template. r. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. s. Double-click the **Object Template** value to display the Single-Choice dialog window. t. Check the checkbox for a different object template, and then click **OK**.  The document template must use an object template. If there is no other object template in this dialog window, you cannot remove the current reference.
Object template	Type definition	u. In the **Type Definitions** folder, select the referencing type definition. v. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. w. Double-click the **Object Template** value to display the Single-Choice dialog window. x. Clear the checkbox for the object template and click **OK**.
Style sheet	Document template	y. In the **Reports and Formatting** folder, double-click the referencing document template to display the Document Template dialog window. z. In the **Select Stylesheet** field, select a different style sheet.  The document template must use a style sheet. If there is no other style sheet in this field, you cannot remove the current reference.

Schema Object	Referencing Object	Action
		aa. Click **Save** to change the style sheet reference.
View	Folder type definition	bb. In the **Type Definitions** folder, select the referencing folder type definition. cc. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. dd. Double-click the **Default View** value to display the Single-Choice dialog window. ee. Clear the checkbox for the view and click **OK**.
View	User	ff. In the **Users** folder, select the referencing user. gg. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. hh. Double-click the **Default View** value to display the Single-Choice dialog window. ii. Clear the checkbox for the view and click **OK**.
View	Folder instance	jj. In the **Where Used** tab, select the referencing folder, and then pull down the **View** menu and choose the **Go To→Go To Object** options. The folder is selected automatically in the Systems Engineering and Requirements Management module. kk. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. ll. Double-click the **Default View** value to display the Single-Choice dialog window. mm. Clear the checkbox for the view and click **OK**.
User	User group	nn. In the **Users** folder, click the plus sign (+) for the user group, and then select the shortcut that represents the user.

Schema Object	Referencing Object	Action
		oo. Pull down the **File** menu, and choose **Delete**. You can also click the **Delete** button on the toolbar or press the delete key.
		A confirmation message asks if you are sure you want to delete the object.
		pp. Click **Yes** to remove the user reference.
User or user group	Security profile	qq. In the **Security Profiles** folder, select the referencing security profile.
		rr. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**.
		ss. Double-click any value that includes the user or user group. Such values may be any or all of the following:
		Change Approvers
		Change Notifiers
		Full Control
		Modify and Read Access
		Read Access
		tt. In the Multi-Choice dialog window, clear the checkbox for the user or user group and click **OK**.
		Repeat steps c. and d. for each remaining value that includes the user or user group.
Security profile	Any other schema object	uu. Open the containing folder and select the referencing schema object.
		vv. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**.
		ww. Double-click the **Security Profile** value to display the Single-Choice dialog window.
		xx. Clear the checkbox for the security profile and click **OK**.

Schema Object	Referencing Object	Action
Security profile	Any object instance	yy. In the **Where Used** tab, select the referencing object, pull down the **View** menu, and choose the **Go To→Go To Object** options. The object is selected automatically in the Systems Engineering and Requirements Management module. zz. Click the **Properties** tab, or display the Edit Properties dialog window by pulling down the **File** menu and choosing **Properties**. aaa. Double-click the **Security Profile** value to display the Single-Choice dialog window. bbb. Clear the checkbox for the security profile and click **OK**.

To remove additional uses of the same schema object, repeat steps 1 and 2.

Procedure Notes

Step 1: You can open the floating window for the **Where Used** tab by clicking the **Open tab** button on the notebook pane toolbar.

Step 2: You can open the floating window for the **Properties** tab by clicking the **Open tab** button on the notebook pane toolbar. To display the Edit Properties dialog window, you can also right-click the object and choose **Properties** from the popup menu.

Step 2: To navigate to an object in the Systems Engineering and Requirements Management module, you can also right-click the object in the **Where Used** tab or window and choose **Go To Object** from the popup menu.

Deleting a Schema Object

In any project except the **TcSE Administration** project, you can delete the following schema objects from the database:

- Activators and macros in the **Activators** folder.
- Property definitions in the **Property Definitions** folder.
- Document templates, object templates, style sheets, and search reports in the **Reports and Formatting** folder.
- Security profiles in the **Security Profiles** folder.
- Object types and subtypes in the **Type Definitions** folder.

User objects cannot be deleted from the database. However, a user can be deactivated to revoke the user's access to all projects. For more information, see Deactivating a User in chapter 5, *Managing Users*.

A given schema object can be deleted only if it is not used by any other object in the project. Before deleting a schema object, you must remove all references to the object. For more information, see Removing References to a Schema Object, earlier in this chapter.

Before deleting a property definition, a type definition, or a Tcl Where clause, ensure that the object is not referenced by a search report object. The report fails if you delete a referenced object without first removing the reference.

The object cannot be recovered.

If groups are deleted using **VDBInspector**, **maintainDB** must be run to clear broken links.

maintainDB can be run by System Administrators only.

To delete a schema object:

1. Open the containing folder, and then select the object in the content table.
2. Pull down the **File** menu, and choose **Delete**.

   A confirmation message is displayed, asking if you want to delete the object.
3. Click **Yes** or press the enter key.

   A second confirmation message is displayed, stating that this operation cannot be undone and asking if you want to continue.
4. To continue, click **Yes** or press the enter key.

   This action clears the queue for the **Undo** option and cannot be reversed.

   The object is removed from the database.

Procedure Notes

Step 2: You can also right-click the object and choose **Delete** from the popup menu. Or, click the **Delete** button on the toolbar.

Deleting a Project

You can delete any project for which you have **Project Administrator** access privilege, whether or not you are the creator of that project.

The project and all of its objects are removed from the database and cannot be recovered.

To delete a project:

1. In the navigation tree or the content table, select the project node.
2. Pull down the **File** menu and choose **Delete**.

   Architect/Requirements displays a confirmation message, asking if you want to delete this object.
3. To continue, click **Yes** or press the enter key.

   Architect/Requirements displays a second confirmation message, stating that a deleted project cannot be restored and asking if you want to continue.
4. To confirm the deletion, click **Yes** or press the enter key.

   This action removes all of the project's data from the database and cannot be reversed.

   The project node and all lower level objects are removed from the Administration module, and the view is refreshed automatically.

   Certain error messages may be generated when objects in the project are displayed in client views immediately before you delete the project. For example, such a message may state that an object is not found or that the selected object does not exist in the database. Regardless of such messages, the project is deleted.

Procedure Notes

Step 2: You can also right-click the project and choose **Delete** from the popup menu. Or, click the **Delete** button on the toolbar.

Chapter 2: Customizing Object Properties

This chapter contains an overview of property definitions and instructions for creating and modifying properties that users can apply to specific objects.

Overview of Property Definitions

All information about the objects in Architect/Requirements is stored in properties. Each property has a property definition that specifies the type of information. There are four different types of properties, and four types of property definitions:

- *Choice* properties have a choice list from which a user can select the value. Project administrators set the list by editing the **Choice List** property of the choice definition. Editing this property launches a dialog window to create or modify the list. Choice lists can be defined as single-choice, which allows only one selection, or as multiple-choice, from which users can select any combination of choices. The **Multiple Choice** property controls this. System-defined choice properties have choice lists that are automatically generated.
- *Date* properties contain a date and time. The **Date Format**, **Time Format**, and **Time Flag** properties of the property definition control the date and time display. Editing any of these properties launches a dialog window to set the format. Date property values reflect the time zone and locale of property definitions in the database, which may differ from the client's time zone and locale.
- *Numeric* properties contain a number. The **Format** property controls how the number is displayed, for example, the number of decimal places. Editing this property launches a dialog window to set the format.
- *Text* properties have values that consist of text entered by the user. The value can be plain text, in a string of unlimited length. Or, the value can be a URL formatted as a hyperlink, through which users can navigate to the URL destination in Microsoft Internet Explorer or Microsoft Windows Explorer.

A fixed set of system-defined properties is always defined for each object type. These property definitions cannot be modified, although users can change the values of some system-defined properties. Other values are system-generated and cannot be modified directly. A project administrator can extend the system-defined properties by creating custom property definitions. The new properties must then by attached to object types before they can be seen in the Systems Engineering module.

To initialize a property's value on new objects, a default value can be set through the **Current Value** property of the property definition. Changing the default value later has no effect on existing objects.

Creating a Property Definition

Architect/Requirements automatically assigns certain property definitions to each built-in object type. The system-defined properties apply to all objects of that type. Although their editable values can be modified, system-defined properties are permanently set for each object type, and individual properties cannot be added or removed. However, you can extend system-defined properties by creating custom property definitions and adding those properties to type definitions.

You can create property definitions at the top level of the **Property Definitions** folder and in subfolders at lower levels. After creating a property definition, you can add the property to any number of different type definitions, including the built-in type definitions. Users can then apply the property to new and existing objects of those types.

You cannot delete a custom property after they are applied to object types.

For more information on applying properties, see Modifying the Properties of a Type Definition in chapter 3, *Customizing Object Types*.

To create a property definition:

1. In the navigation tree or the content table, open the parent folder for the new property definition.

   The content table displays the existing property definitions in the folder.

2. Pull down the **File** menu and choose one of the following:

   - For a choice property, choose the **New→Property Definition→Choice** options.
   - For a date property, choose the **New→Property Definition→Date** options.
   - For a numeric property, choose the **New→Property Definition→Numeric** options.
   - For a text property, choose the **New→Property Definition→Text** options.

   The content table displays the new property definition with a default name in an open text field.

3. Enter the property name, and then press the enter key.

   To place the property definition according to the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the root node and choose **Refresh** from the popup menu.

Procedure Notes

Step 1: To open subfolders, click the plus signs, or double-click a folder with a plus sign. In the navigation tree, you can also select a folder and then pull down the **View** menu and choose **Expand**, or right-click the folder and choose **Expand** from the popup menu. In the content table, you can also select a folder and then pull down the **View** menu and choose **Expand All**, or right-click the folder and choose **Expand All** from the popup menu.

Step 2: You can also right-click the parent folder in the navigation tree, or right-click anywhere in the content table, and then choose the options from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

The notebook views do not respond when the **Property Definitions** folder, or one of its sub-folders, is selected in the administration module's navigation pane. The **Properties and Attachments** notebook tab displays information from the previously selected object and not for

the selected **Property Definitions** folder. The notebook tabs responds when the **Property Definitions** folder is selected in the content pane.

To view the properties or notes of a properties definition folder, select the folder in the content pane.

Modifying a Choice Property Definition

A choice property has a list of choices from which users select the property value in the Systems Engineering module. By modifying a choice property definition, you can define the choice list type as single-choice, from which users can select only one value, or as multiple-choice, from which users can select any combination of choices as the value. You can add choices, rename choices, change the list order, and set the default value for the property. Also, you can delete any number of choices from the list.

System-defined choice properties have choice lists that are automatically generated.

To modify a choice property definition:

1. In the **Property Definitions** folder, select the choice definition, or open the containing subfolder and then select the choice definition.

   The **Properties** tab or window displays all viewable properties for the choice definition.

2. In the **Value** column, double-click the value for the **Choice List** property, the **Current Value** property, or the **Multiple Choice** property.

   The Modify Choice Property Definition dialog window is displayed, with the current choices in the **Choice List** pane.

3. Do any or all of the following:
   - To define the type of choice list:
     - For a list that allows only one selection, click **Single Choice**.
     - For a list that allows any number of selections, click **Multiple Choice**.

     If you change an existing list from multiple-choice to single-choice, the change becomes effective when a user edits a value for that property in the Systems Engineering and Requirements Management module.

   - To add choices:
     - Do one of the following with **Choices** selected:
       - To add a choice in the position directly above an existing choice, select the existing choice in the list, and then click the **Add** button.
       - To add a choice at the bottom of the list, clear any current selection by holding down the control key and clicking the selection, and then click the **Add** button.

The Add dialog window is displayed.

- Enter the new choice in the text field, and then click **OK**.

  The choice is added to the list and is cleared from the text field. You can enter another new choice in the text field, and then click **OK** to add that choice and clear the field again. When you are finished adding choices, click **OK** with the text field cleared to close the Add dialog window.

You can continue adding choices by repeating steps a and b.

- To rename a choice, do the following with **Choices** selected:

  -

    Select the choice in the list, and then click the **Rename** button.

    The Rename dialog window is displayed.

  - Enter the name in the text field, and then click **OK** to close the dialog window.

Consider how a name change may affect the choice's current use within your project. For example, database synchronization problems may occur if the old name is used in existing object templates, live Excel spreadsheets, or live Visio diagrams. Also, search results may not be returned by saved searches that specify the old name in the criteria.

- To change the list order, do any of the following with **Choices** selected:
  - To move a choice up in the list, select the choice, and then click the **Move Up** button until the choice reaches the intended position.
  - To move a choice down in the list, select the choice, and then click the **Move Down** button until the choice reaches the intended position.
  - To sort the choices in alphabetical order, click the **Sort List** button.
- To set the default value:
  - Click **Defaults**.

    At the left of the choices in the list, buttons are displayed for a single-choice list or checkboxes are displayed for a multichoice list.

  - Do one of the following:
    - For a single-choice list, click the button for the choice that you want to set as the default value.
    - For a multi-choice list, check the checkbox for each choice that you want to add to the default value, and clear the checkbox for each choice that you want to remove from the default value.

You can remove the entire default value by clicking **Clear**. However, this action is not allowed if the choice definition's **Input Required** property value is **Yes**. To set the value to **No**, you must first close the Modify Choice Property Definition dialog window.

- To delete a choice, select it in the list, and then click the **Delete** button.

The **Delete** button is unavailable if the selected choice is included in the default value. After you remove the choice from the default value, the button is available and you can delete the choice.

The deleted choice is marked with arrow brackets (< >) and highlighted. The selection moves to the next lower choice in the list, or the deleted choice remains selected if it occupies the last position.

You can reverse this action until you close the dialog window. To change the **Delete** button label to **Undelete**, select the deleted choice in the list. Then, click **Undelete** to remove the arrow brackets and highlighting from the choice.

A choice cannot be deleted if it currently appears in the property value for any object. This condition applies to all existing objects, including those in users' Recycle Bins.

- If the choice is used only by objects that are not irreversibly frozen, the choice must first be removed from all occurrences of the property value. Then, the choice can be deleted.
- If the choice is used by a baselined object or a superseded frozen object, the property value cannot be edited and the choice cannot be deleted.

Database synchronization problems may occur if a deleted choice is used in existing object templates, live Excel spreadsheets, or live Visio diagrams. Search results may not be returned by saved searches that specify a deleted choice in the criteria.

4. Click **OK** to close the dialog window.
   - If you did not delete a choice, the **Properties** tab or window displays your changes. You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.
   - If you deleted a choice, a confirmation message states that this operation cannot be undone and asks if you want to continue.
     - To apply your changes in the **Properties** tab or window, click **Yes**.

       This action clears the queue for the **Undo** option and cannot be reversed.

     - To cancel all changes, click **No**.

To continue modifying the choice definition, you can specify that the value must contain at least one choice in the Systems Engineering module. First, set a default value for the choice definition as described in step 4-3. Then, double-click the **Input Required** property value to display the Single-Choice dialog window, and check the **Yes** checkbox to require that a value must be entered.

The **No** checkbox is checked by default, which allows the choice property value to be blank and allows users to clear existing choices. When the **Input Required** property is changed from **No** to **Yes**, the value of each occurrence of the choice property is retained in the Systems Engineering module. The next time each value is edited, at least one choice is required. Otherwise, the previous value is retained and an error message is displayed.

Procedure Notes

Step 1: To open subfolders, click the plus signs, or double-click a folder with a plus sign. You can also select a folder and then pull down the **View** menu and choose **Expand** or **Expand All**. Or, right-click the folder and choose **Expand** or **Expand All** from the popup menu.

Setting a Dynamic Choice List for a Choice Property Definition

The choice list of a choice property definition can be updated dynamically through a system-defined activator subtype, **Picklist**. Activators are used to execute scripts written in Tool Command Language (Tcl). When a **Picklist** activator is applied to a choice definition, the script result automatically populates the choice definition's **Choice List** property value.

For example, assume that a choice definition named **Assign To** has a choice list of the users in your project. Also, assume that a **Picklist** activator named **Update Users** contains the following Tcl script:

```
set project $currentProject
set members [getList $project USER_LIST]
```

In this example, the script returns a list of all users in the project. When the **Update Users** activator is applied to the **Assign To** choice definition, the choice list is automatically modified when a user is added, renamed, or removed.

The choice definition can be either single-choice or multiple-choice. By applying the choice definition to a type definition, changes to the choice list are made immediately available for selection on those objects in the Systems Engineering module.

- The **Choice List** value cannot be edited manually when a **Picklist** activator is applied to the choice definition. All choices are set automatically by the activator script result.
- A default value cannot be set for the choice definition.
- The choice list cannot be rearranged or sorted.

To create a Picklist activator:

1. Select the **Activators** folder or subfolder where you want to create the activator.
2. Pull down the **File** menu and choose **New→Activator Subtype→Picklist**.

   The content table displays the activator with a default name in an open text field.
3. Enter the activator name, and then press the enter key.

The desired Tcl script must be coded in the activator. Sample scripts are provided in the following **Picklist** activators:

- **Choice List** returns a list of the folders in the project.
- **User** returns a list of the users in the project.
- **User Group** returns a list of the users and user groups in the project.

These **Picklist** activators are included in the **Activators** folder of the Systems Architect/Requirements Management Administration project. This makes them available for use in any project.You can open a script in Microsoft Notepad by selecting the activator, pulling down the **File** menu, and choosing **Open**. Or, you can double-click the **Script** value in the **Properties** tab to open the script in a text field. For more information about using activators, see the *Systems Architect/Requirements Management API Reference* manual.

Procedure Notes

Step 2: In the navigation tree, you can also right-click the folder and choose **New→Activator Subtype→Picklist** from the popup menu. In the content table, you can also right-click any object and choose **New→Picklist** from the popup menu.

To apply a Picklist activator to a choice property definition:

- This procedure clears all instances of the property value if the property is used by existing objects and if a **Picklist** activator is not already applied.
- If you change from one **Picklist** activator to another, existing objects that use the property retain their current values. The new activator's choice list is available the next time each property instance is edited.

1. In the **Property Definitions** folder or a subfolder, select the choice definition to display its properties in the **Properties** tab.
2. In the **Value** column, double-click the value for the **Update Choice List Dynamically** property.

   The Single-Choice dialog window displays the **Picklist** activators in the project.

   The following **Picklist** activators are included by default:

   - **Choice List** returns a list of the folders in the project.
   - **User** returns a list of the users in the project.
   - **User Group** returns a list of the users and user groups in the project.

3. Check the checkbox for the activator, and then click **OK**.

   The script result populates the **Choice List** property value.

Procedure Notes

Step 1: Instead of the **Properties** tab, you can use the **Properties** floating window by clicking the **Open tab** button on the notebook pane toolbar. Or, pull down the **File** menu and choose **Properties** to use the Edit Properties dialog window.

To remove a Picklist activator from a choice property definition:

This procedure clears all instances of the property value where the property is used by existing objects.

1. In the **Property Definitions** folder or a subfolder, select the choice definition to display its properties in the **Properties** tab.
2. In the **Value** column, double-click the value for the **Update Choice List Dynamically** property.

   The Single-Choice dialog window displays the **Picklist** activators in the project.

3. Clear the checkbox for the activator, and then click **OK**.

Procedure Notes

Step 1: Instead of the **Properties** tab, you can use the **Properties** floating window by clicking the **Open tab** button on the notebook pane toolbar. Or, pull down the **File** menu and choose **Properties** to use the Edit Properties dialog window.

To apply a choice property definition to a type definition:

1. In the **Type Definitions** folder or a subfolder, select the type definition to display its properties in the **Properties** tab.
2. In the **Value** column, double-click the value for the **Properties** property.

   The Multi-Choice dialog window displays the properties in the project.
3. Check the checkbox for each property to apply, and then click **OK**.

Step 1: Instead of the **Properties** tab, you can use the **Properties** floating window by clicking the **Open tab** button on the notebook pane toolbar. Or, pull down the **File** menu and choose **Properties** to use the Edit Properties dialog window.

Modifying a Date Property Definition

You modify a date definition to set the property value's valid format and default value. When a user double-clicks the property value in the Systems Engineering module, the valid format appears below the text field.

To modify a date property definition:

1. In the **Property Definitions** folder, select the date definition, or open the containing subfolder and then select the date definition.

   The **Properties** tab or window displays all viewable properties for the date definition.

2. In the **Value** column, double-click the value for the **Current Value**, **Date Format**, **Time Flag**, or **Time Format** property.

   The Modify Date Property Definition dialog window is displayed.

3. Do any or all of the following:

   - To set the valid format for the date, click one of the choices under **Choose a date format**.
   - To set the valid format for the time of day, do one of the following:
     - Click one of the choices under **Choose a time format**.
     - Clear the **Including Time** checkbox.

       This action specifies that the format does not include the time of day.
   -

     The default value for date properties can be set to **Today**. Setting the default value to **Today** initializes instance of the property to the day they are created. If **Include Time** is selected, the time of creation is also set.

     If **Include Time** is not selected, only the date is displayed on the property instances. However, a time value is still stored in the database. The time for such property instances is initialized to noon GMT. This provides consistency and prevents time zone adjustments from shifting the time to a different day. If **Include Time** is selected later, the noon GMT times becomes visible for existing property instances. The time is adjusted to the local time zone when displayed.

     To set the default value of the date definition, do one of the following:

     - In the text field at the bottom of the dialog window, delete the current value and enter the new value.

       The format must match the selected date format and time format.
     -

       To set the value to today's date, click the **Today** button.
     -

       To mark the value for setting at a later date, click the **TBD** button.

4. Click **OK**.

The **Properties** tab displays your entries.

If you set **TBD** as the default value, each instance of the property value is **TBD** until the instance value is edited or the default value is changed.

Procedure Notes

Step 4: To reverse this action, you can pull down the **Edit** menu and choose **Undo Change Properties**, click the **Undo** button on the toolbar, or press control-Z.

Modifying a Numeric Property Definition

Modify a numeric definition to set the valid format, default value, and formula for calculation. When a user double-clicks the value in the Systems Engineering module, the valid format is shown below the text field. When a user calculates the value for a selected parent object, the formula operates on the value for each direct child.

To modify a numeric property definition:

1. In the **Property Definitions** folder, select the numeric definition, or open the containing subfolder and then select the numeric definition.
2. In the **Value** column of the **Properties** tab or window, double-click the value for the **Current Value**, **Format**, or **Formula** property.
3. In the Modify Numeric Property Definition dialog window, do any or all of the following:
   - To set the valid format, select **Normal**, **Percentage**, or **Scientific Notation** under **Choose display format**.
   - In the **Decimal Places** field, enter the number of digits allowed to the right of the decimal point.
   - To set the default value, enter the value in the **Enter default value** field.

     The default value format must match the selection under **Choose display format**.
   - To use a formula for calculating the property value automatically, select one of the following in the **Formula** field:

     Average calculates the sum of the values divided by the number of direct children.

     Maximum calculates which value is the highest among all direct children.

     Minimum calculates which value is the lowest among all direct children.

     Multiply calculates the product of the values for all direct children.

     Sum calculates the total of the values for all direct children.

     For more information about calculating numeric property values, see the *Systems Architect/Requirements Management User's Manual*.
4. Click **OK** to close the dialog window and apply your changes.

Procedure Notes

Step 1: To open subfolders, click the plus signs, or double-click a folder. In the navigation tree, you can select a folder and then pull down the **View** menu and choose **Expand**, or right-click the folder and

choose **Expand** from the popup menu. In the content table, you can select a folder and then pull down the **View** menu and choose **Expand All**, or right-click the folder and choose **Expand All** from the popup menu.

Step 4: To reverse this action, you can pull down the **Edit** menu and choose **Undo Change Properties**, click the **Undo** button on the toolbar, or press control-Z.

Modifying a Text Property Definition

A text property value consists of text that users enter in the Systems Engineering module. By modifying the corresponding text property definition, you can define the type of text value and you can set the default value for the property.

The value can contain text of one of the following types:

- Plain text, the default text type. There is no limit on the length of the text string. Users can enter different text strings for individual objects where the property applies in the Systems Engineering module.
-

  Hypertext, for example, a URL for a Web address or a network file location. In the Systems Engineering module, the value is formatted as a hyperlink in the content table and the **Properties** tab or window. By holding down the control key and clicking the hyperlink, users can navigate to the URL destination in Microsoft Internet Explorer or Microsoft Windows Explorer.

  You can include the property value in Microsoft Excel and Microsoft Word export files for hyperlink navigation to the URL destination. For more information, see Modifying an Object Template and Modifying an Excel Template in chapter 4, *Customizing the Interface With Microsoft Office*.

  A unique URL can be entered for each object where the property applies in the Systems Engineering and Requirements Management module. Hyperlink navigation is disabled while the URL is edited.

  The URL is not checked for validity. Therefore, users must ensure that the URL is correct and that it conforms to syntax requirements. For example, a file location must be entered in the long UNC path format. Also for example, a Web address may require a preface such as **http://**, **https://**, or **ftp://**.

For both text types, you can specify a default value or leave the default value blank.

To modify a text property definition:

1. In the **Property Definitions** folder, select the text definition, or open the containing subfolder and then select the text definition.

   The **Properties** tab or window displays all viewable properties for the text definition.

2. Do one or both of the following:
   - To define the type of text value:
     .

       In the **Value** column, double-click the value for the **Content Type** property to display the Single-Choice dialog window.

     . Do one of the following:
       - For plain text, check the **String** checkbox.
       - For hypertext, check the **URL** checkbox.

     . Click **OK** to close the dialog window and set the value type.

- To set the default value:

  .

  In the **Value** column, double-click the value for the **Current Value** property.

  The Modify Text Property Definition dialog window is displayed.

  . Enter the default value in the text field, and then click **OK** to close the dialog window and set the property value.

  - For a hypertext default value, ensure that the URL is correct and that it conforms to syntax requirements. A file location must be entered in the long UNC path format. Also, a Web address may require a preface such as **http://**, **https://**, or **ftp://**.
  - The value for the **Current Value** property is not formatted as a hyperlink in the Administration module.

  You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Procedure Notes

Step 1: To open subfolders, click the plus signs, or double-click a folder with a plus sign. In the navigation tree, you can also select a folder with a plus sign and then pull down the **View** menu and choose **Expand**, or right-click the folder and choose **Expand** from the popup menu. In the content table, you can also select a folder with a plus sign and then pull down the **View** menu and choose **Expand All**, or right-click the folder and choose **Expand All** from the popup menu.

Step 2: In setting the default value, you can enter new text, and you can change or delete any existing text.

Updates to Change Time and Change User Properties

The **Change Time** and the **Change User** properties record only the significant changes to an object. It is not intended to record every change to an object. The significance of changes varies by the object type. The business processes built around the Architect/Requirements also have an impact on what is considered significant. However, the rules cannot be configured.

The **Change Time** and the **Change User** properties are not updated by the following events:

- Setting of most properties, including **Name**.
- Adding or removing a child of a requirement or a building block.
- Moving an object to a new owner, by dragging it or by **Cut** and **Paste**.

  Neither the moved object nor the new owner is updated, except the folders are updated when a member is added/removed.
- Adding or remove an attachment, such as a note or a diagram.

Table 2-1 lists the triggering events that cause the **Change Time** and the **Change User** properties to be updated. These triggering events vary by the object type.

These triggering events may change from time to time, as new functionalities are added, and based on the customer feedback.

Table 2-1. Triggering Events for Updates to Change Time and Change User Properties

Object Type	Triggering Events
All objects	• Add/Remove a defining or a complying trace link. • Delete or move to the recycle bin. • Restore from the recycle bin. • Change subtype. • When passing through the change management process and setting. It includes events such as users to approve, users notified, users already approved, and users rejected.
Folder	• Add/Remove a member. • Set the icon overlay. • Set the document template.
Requirement and **Paragraph**	• Edit the text. • Freeze/Unfreeze/Baseline. • Set the text format property.
Building Block	• Freeze/Unfreeze/Baseline.

Table 2-1. Triggering Events for Updates to Change Time and Change User Properties

Object Type	Triggering Events
	• Set the numbering property (not the number).
Note	• Edit the text. • Set the text format property.
Diagram	• Save. • Set the data dictionary.
Connection	• Set the data definition.
Port	• Set the data definition. • Set the direction.
Spreadsheet	• Save. • Set the file type.

Chapter 3: Customizing Object Types

This chapter contains an overview of type definitions and instructions for using subtypes to extend built-in object types.

Overview of Type Definitions

A type definition specifies the properties that apply to all objects of a given type. In turn, these properties determine the nature and behavior of the related objects.

For every new project, Architect/Requirements generates type definitions automatically in the Administration module. All projects receive the same type definitions, named **Folder**, **Requirement**, **Building Block**, **Group**, **Note**, **Trace Link**, **User Group**, **Connection**, **Diagram**, **Spreadsheet**, **Change Approval**, and **Change Log**. Through particular system-defined properties, each of these type definitions governs a built-in object type, or *base type*.

The project administrator can customize a base type by modifying the properties of the related type definition. Default values can be changed for editable system-defined properties, and user-defined properties can be applied to the base type definition.

By creating custom type definitions, or *subtypes*, the project administrator can extend the built-in object types for specialized purposes. From an existing type definition, any number of subtypes can be created for a given object type. Different properties and behavior can be associated with each subtype according to its purpose.

A subtype inherits the base type of the originating type definition, or parent, and also inherits the parent's system-defined and user-defined properties. Though the base type cannot be changed, a subtype can be distinguished by unique properties appropriate to its purpose. The project administrator can create specific user-defined properties for the subtype and add them to its property set. Inherited user-defined properties can be modified or removed to fit the purpose. System-defined properties also can be modified, if editable, but they cannot be added or removed.

Subtypes can be created from base type definitions and from other subtypes. The content table shows the subtype hierarchy in the **Type Definitions** folder. Once a subtype is created in the Administration module, users can assign the subtype when creating new objects of that base type in the Systems Engineering module. Users can assign the new subtype to existing objects by editing the **Subtype** properties. A system-defined folder subtype (**Document**), a system-defined requirement subtype (**Paragraph**), and a system-defined building block subtype (**TRAM**), can be assigned to new and existing objects of the related type.

Customized type definitions are specific to the project. Other projects in the same Architect/Requirements installation may be customized for different purposes, processes, or products.

Creating a Subtype

In each new project, Architect/Requirements automatically creates type definitions for the built-in object types. To customize those object types, you create subtypes, which are custom type definitions based on the system-defined type definitions.

Subtypes allow you to apply different properties among built-in objects of the same type. For example, you may create requirement subtypes named Customer Requirements and Derived Requirements, to distinguish requirements by specific purposes. Then you would apply a unique set of user-defined properties to each new subtype. For more information, see Modifying the Properties of a Type Definition, later in this chapter, and chapter 2, **Unsatisfied xref title**.

Each subtype inherits the system-defined properties of the parent type definition on which the subtype is based. For more information about system-defined properties in the Systems Engineering module, see the *Systems Architect/Requirements Management User's Manual*.

In addition, each subtype inherits the security profile named in the **Instance Security Profile** property of the parent type definition. For more information, see Assigning a Default Security Profile to New Objects of a Type in chapter 6, *Maintaining Project Security*.

After you create the subtype, users can assign it to new objects of that base type in the Systems Engineering module. For existing objects, users can assign the new subtype by changing the **Subtype** property values.

To create a subtype:

1. In the navigation tree or the content table, open the **Type Definitions** folder for the project.

   The content table displays the base type definitions. In the **Types/SubTypes** column, a plus sign (+) is shown for each parent type definition with one or more subtypes at lower levels. To see lower level subtypes, click the plus signs.

2. Select the parent type definition for the new subtype, pull down the **File** menu, and choose the **New→Subtype** options.

   The content table displays the new subtype at the next level below the parent, with a default name in an open text field.

3. Enter the subtype name, and then press the enter key.

Procedure Notes

Step 2: You can also right-click the parent and choose **New Subtype** from the popup menu.

Modifying the Properties of a Type Definition

After creating a subtype, you can apply a unique set of user-defined properties to customize the object type. When users create those objects in the Systems Engineering module, the new objects receive the properties that you apply to the type definition.

All existing objects of the type are updated with the changes. The extent of this transaction depends on the number of objects that are updated.

If a large number of objects is involved, Siemens PLM Software recommends one of the following:

- Perform this procedure during times of low system usage.
-

The Architect/Requirements system administrator set the **MessageQueue.Start** parameter for background processing. For more information about the **MessageQueue.Start** parameter, see the *Systems Architect/Requirements Management System Administrator's Manual*.

For example, a custom object template can be assigned to each type definition through its **Object Template** property. Also, a default security profile can be assigned to new objects through the **Instance Security Profile** property of the type definition. For more information, see Object Templates in chapter 4, *Customizing the Interface With Microsoft Office*, and Assigning a Default Security Profile to New Objects of a Type in chapter 6, *Maintaining Project Security*.

User-defined properties can be added and removed for any type definition, including the base type definitions. System-defined properties cannot be added or removed.

To modify the properties of a type definition:

1. In the navigation tree or the content table, open the **Type Definitions** folder.

   The content table displays the base type definitions. To see lower level subtypes in a hierarchy, click the plus signs in the **Types/SubTypes** column.

2. In the content table, select the type definition.

   The properties table displays all viewable properties for the type definition.

3.

   In the **Value** column, double-click the **Properties** value.

   Depending on the property type, the Single-Choice or Multi-Choice dialog window is displayed.

4. Do one or both of the following:

   - To add an unapplied property, check the checkbox.
   - To remove an applied property, clear the checkbox.

     This action disables all existing reference links to all instances of the property. Each disabled reference link is marked with the label **[Broken Link:** *contents***]**. The *contents* variable represents the last actual value of the property before the reference link was disabled. For more information about reference links, see the *Systems Architect/Requirements Management User's Manual*.

In the Multi-Choice dialog window, you can click **Select All** to add all properties simultaneously. Or, click **Unselect All** to remove all properties simultaneously.

5. Click **OK**, or press the enter key.

   A confirmation message states that this operation updates all instances of this object type.

   The updates are processed in the background if the **MessageQueue.Start** parameter is set to **true**.

6. To continue, click **Yes**, or press the enter key.

   The changes are shown in the **Properties** value for the type definition. Architect/Requirements processes all instances of the object type and applies the updates.

   If a large number of objects are involved, you can perform other actions while the updates are processed in the background. A message is displayed when updates are complete. Any instances where the update process failed are indicated.

   Property updates may take a few seconds to many minutes, depending on the number of objects of that type and how often the background queue is set to run. Even when there are only a few objects to be updated, it may be a minute or more before the queued background update begins.

   The objects are collected for update using a query, so all objects in a folder may not be processed at the same time. Update of some objects may also be delayed if users are modifying them at the time they are first processed.

   As a result, users may open a folder and see some objects that have the property changes applied and other objects where the old properties are still in effect.

Procedure Notes

Step 6: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

The **Undo** action is applied immediately to the **Type Definition's Properties** list. However, individual objects of that type are updated in the background as described above. These updates may take some time to complete, so some objects reflect the **Undo** action before others.

Customizing the Object Type Indicator for a Type Definition

You can modify the object type indicator with a custom icon for type definitions (and their subtypes) in the Administration module.

The recommended graphic size is 16 pixels by 16 pixels, which is also the smallest allowable graphic size. Larger icons can be used (up to 100 pixels x 100 pixels), however, their heights are clipped to the row height of the view they appear in. Also, the placements of the icon overlays, such as **Frozen** and **Shortcut**, are placed assuming a height of 16 pixels, and not the actual height.

The object type indicator for a selected child object is determined by the value of the child's **Icon** property:

- If the **Icon** value is the default graphic, the child inherits its object type indicator from its parent. When the parent's graphic is changed, that same graphic is automatically applied to the child.
- If the **Icon** value is a graphic other than the default, the child retains that object type indicator regardless of changes to the parent's graphic. When the child's graphic is changed, the graphic for the parent is not affected.

To customize the object type indicator for a type definition:

1. In the navigation tree or the content table, open **Type Definitions** folder for the project.

   The content table displays the base type definitions. To see lower level subtypes in a hierarchy, click the plus signs in the **Types/SubTypes** column.

2. In the content table, select the type definition.

   The properties table displays all viewable properties for the type definition.

3. In the **Value** column, double-click the **Icon** value.

   Architect/Requirements displays the **Open** dialog window, which lists existing graphic files in the current drive or folder. If the list does not display the graphic that you want, use the **Look in** field to change the drive or folder, or use the **Files of type** field to display other types of files.

4. Select the graphic, and then click **OK** to close the dialog window.

   The new graphic is displayed in the **Icon** property value.

Setting the Default View for a Folder Type Definition

In the Systems Engineering and Requirements Management module, users can apply saved views to folders selected in the navigation tree. The view for each folder determines the display of property columns in the content table. The default view that you set for a folder type definition determines the initial view for new and existing folders based on that type definition.

A user can locally modify the default view by adding, removing, resizing, or sorting columns. Modifications are recorded on the user's computer and persist until the user reverts to the default view. Also, a user can select a different view in the **View** field, and the selection is recorded locally.

- When you create a folder type definition, it inherits the default view of the parent type definition. You can then set a different default view. For more information, see Creating a Subtype, earlier in this chapter.
- An existing subtype's default view is not affected if you set a different default view for the parent.

To set the default view for a folder type definition:

1. Make the view visible to all users in the project by doing the following:
    a. In the **Reports and Formatting** folder, select the view object, and then click the **Properties** tab to display the view's properties.
    b. In the **Value** column, double-click the **Shared State** value.
    c. In the Single-Choice dialog window, check the **Public** checkbox and click **OK**.
2. In the **Type Definitions** folder, select the folder type definition.

    The **Properties** tab displays the type definition's properties.
3. In the **Value** column, double-click the **Default View** value.
4. In the Single-Choice dialog window, do one of the following:
    a. To specify a default view, check the checkbox for the view and click **OK**.
    b. To remove the current default view setting, clear the checkbox and click **OK**.

While a view is set as the default, its **Shared State** property value cannot be changed. Furthermore, a view object cannot be deleted if it is the default view for a folder type definition, a folder , or a user object. Before changing the **Shared State** value or deleting the view , you must remove all uses of the view by other objects. For more information, see Removing References to a Schema Object in chapter **Unsatisfied xref reference**, ***Unsatisfied xref title***, and appendix System-Defined Properties in the Administration Module.

Procedure Notes

Steps 1 and 2: Instead of the **Properties** tab, you can use the **Properties** floating window by clicking the **Open tab** button on the notebook pane toolbar. Or, pull down the **File** menu and choose **Properties** to use the Edit Properties dialog window.

Customizing the ROIN for a Requirement Type Definition

A Requirement Object Identification Number (ROIN) identifies each requirement within a project. The **ROIN** property, a read-only system-defined property, displays the ROINs in the Systems Engineering and Requirements Management module, providing a continuous numbering scheme. In the Administration module, you can assign custom prefixes and formatting to ROINs through the **ROIN Counter** property of requirement subtypes.

- ROINs assigned by Architect/Requirements are unique within a project.
- Custom ROINs can be duplicated within a project. Architect/Requirements does not check for duplicate prefixes or numbers.

An editable text property, the **ROIN Counter** property specifies the following:

- An alphanumeric prefix for the ROIN.
- The minimum number of digits in the numeric portion of the ROIN.

  Numbers with less than the minimum digits are padded with leading zeros.
- A starting number.

For example, **REQ-0100** specifies the following:

- **REQ-** as the prefix.
- A minimum of four digits in the numeric portion.
- **100** as the starting number.

The **Requirement** base type has a default **ROIN Counter** value of **0001**, with no prefix, four minimum digits, and a starting number of **1**. When a subtype is created, its **ROIN Counter** value is blank by default. A blank value indicates that the subtype uses the base type counter.

Though all subtypes can share the base type counter, you can customize the ROIN for a subtype by changing the subtype's **ROIN Counter** property value. Thereafter, the subtype uses the new counter and cannot revert to the base type counter.

The subtype's current **ROIN Counter** value becomes the **ROIN** property value for the next new requirement of that subtype. When the requirement is created, the **ROIN Counter** value increases by one, for example, from **REQ-0100** to **REQ-0101**.

Siemens PLM Software recommends that you customize ROINs for all requirement subtypes before any requirements are created, and that you do not subsequently change the counters. However, **ROIN Counter** properties can be changed without affecting the ROINs of existing requirements. Such changes apply only to new requirements.

Architect/Requirements does not prevent **ROIN Counter** properties from being changed in ways that produce duplicate ROINs, such as assigning the same prefix to different subtypes. The project administrator is responsible for ensuring that ROINs remain unique within a project. To keep custom ROINs unique, do not change the prefix or the subytpe.

To customize the ROIN for a requirement type definition:

1. In the navigation tree or the content table, open the **Type Definitions** folder.

   The content table displays the base type definitions. To see other requirement subtypes in the hierarchy, click the plus sign to the left of the **Requirement** type definition and plus signs at lower levels.

2. In the content table, select the type definition.

   The properties table displays all viewable properties for the type definition.

3. In the **Value** column of the properties table, double-click the **ROIN Counter** property value.

   A text field opens in the cell.

4. Enter the custom prefix and format in the text field, and then press the enter key.

   You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Chapter 4: Customizing the Interface With Microsoft Office

This chapter provides overviews and instructions for using formatting objects to customize data exported to Microsoft Office Word, Microsoft Office Excel, and Microsoft Office Visio.

Templates and Style Sheets for Export to Microsoft Office Word

In the Systems Engineering and Requirements Management module, project users can export object data from the database to Microsoft Office Word. Using a *document template* as a reference, Architect/Requirements generates a Word document containing data for each object that the user specifies.

The document template controls the export process through the following:

- Two *object templates*, each of which is assigned to several type definitions and determines the data that is exported for those object types.
- A *style sheet*, which determines the Word formatting that is applied to the data.

In the Administration module, several document templates, object templates, and style sheets are created automatically in the **Reports and Formatting** folder of every new project. The project administrator uses Word to work with object templates and style sheets. For more information about exporting objects to Microsoft Office Word, see the *Systems Architect/Requirements Management User's Manual*.

Document Templates

Architect/Requirements references a document template in generating each export document. In turn, the document template supplies instructions through the object templates and the style sheet that are associated with the document template.

Document templates can be used to customize the export process for a project. For example, the project administrator can modify the **Default Text Object Template**, the **Default Object Template**, and the **Default Style Sheet**, which are automatically associated with the **Default Document Template** in a new project.

The default document template is overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up a customized default template from the earlier version before upgrading to a later version.

Also, custom object templates can be assigned to type definitions through the **Default Document Template** or a custom document template, and custom style sheets can be associated with any document template. Users can choose different combinations of data and formatting for each export document. For more information, see Creating a Document Template and Modifying a Document Template, later in this chapter.

Creating a Document Template

By its associated object templates and style sheet, a document template controls the Microsoft Office Word export process. You can customize that process for your project through unique document templates with object templates and style sheets that you create.

The **Default Text Object Template**, the **Default Object Template**, and the **Default Style Sheet** are automatically associated with the new document template. After creating the document template, you can change the object template assigned to one or more type definitions, and you can associate a different style sheet. For more information, see Modifying a Document Template, Object Templates, and Style Sheets, later in this chapter.

Once the document template is created in the Administration module, users can choose the document template when exporting object data in the Systems Engineering and Requirements Management module.

To create a document template:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Pull down the **File** menu and choose the **New→Document Template** options.

   The content table displays the document template with a default name in an open text field.

3. Enter the template name, and then press the enter key.

   To refresh the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the **Reports and Formatting** folder and choose **Refresh** from the popup menu. You can also click the **Refresh** button on the toolbar.

Procedure Notes

Step 2: You can also right-click the **Reports and Formatting** folder and choose the **New→Document Template** options from the popup menu. Or, right-click in the content table and choose **New→Template→Document Template** from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

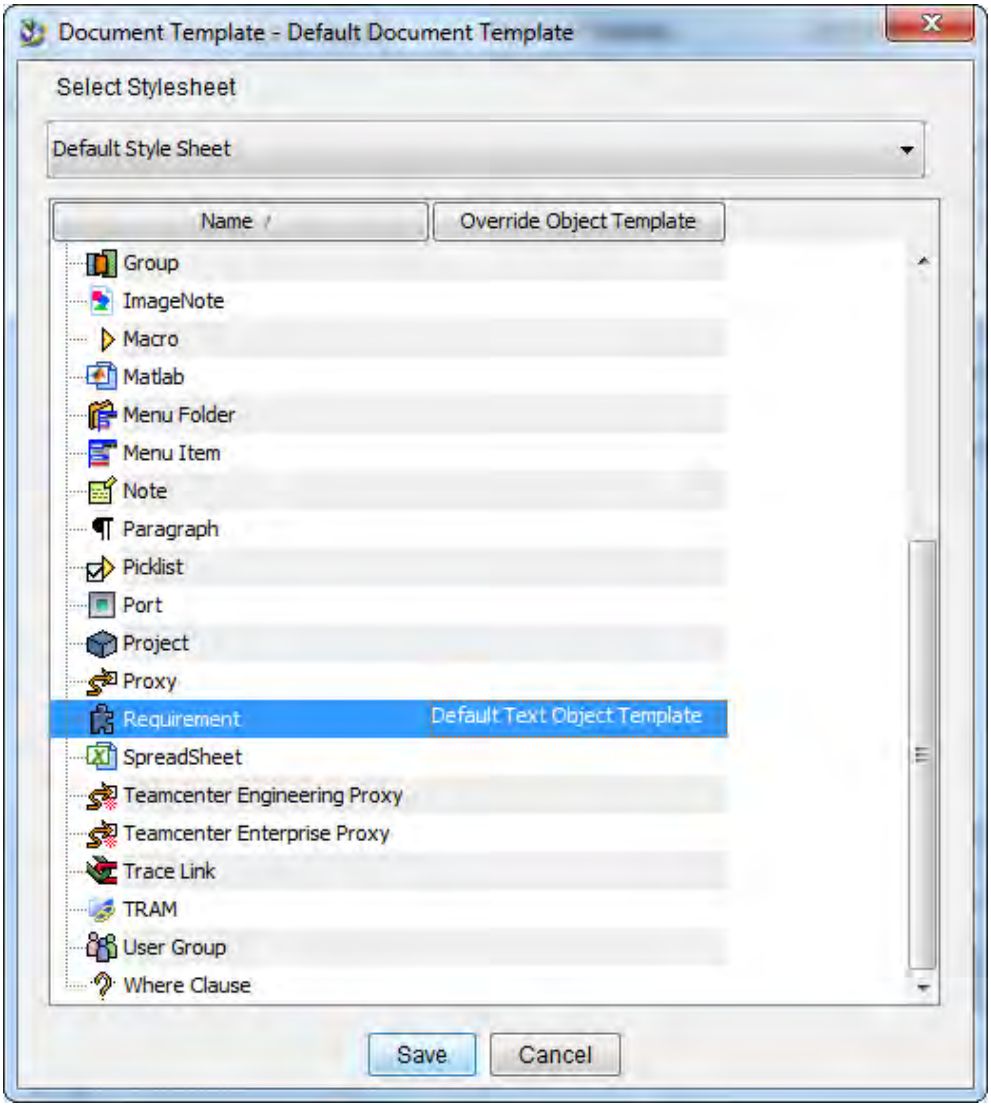
Modifying a Document Template

When you create a document template, the **Default Text Object Template**, the **Default Object Template**, and the **Default Style Sheet** are associated automatically. You can modify those associations to customize the new document template for a specific purpose in your export process.

The default document template is overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up a customized default template from the earlier version before upgrading to a later version.

If you use the **Object Template** property for requirements, they are overridden by the **Default Template** settings.

If the **Default Document Template** has the **Override Object Template** for Requirement set to **Default Object Text Template**, it overrides the object template property setting.



If the **Default Document Template** contains an override for the object, the **Default Document Template** is used for the preview. This is irrespective of the **Object Template** that you set for a **Type Definition**.

For built-in type definitions and custom subtypes, you can assign object templates that you create to export different data for different object types. Also, you can associate a custom style sheet with the new document template. In addition, you can modify the **Default Document Template** to use custom object templates and a custom style sheet. For more information, see Creating a Document Template, earlier in this chapter, and Object Templates and Style Sheets, later in this chapter.

To modify a document template:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays all templates, style sheets, and reports in the project.

2.

   Select the document template, pull down the **File** menu, and choose **Open**.

   The Document Template dialog window is displayed.

3. Do any of the following:

   - To change the style sheet for the document template, select the new style sheet in the **Select Stylesheet** field.
   -

     To change the object template for a type definition in the **SubTypes** column:

     . Double-click the cell in the **Override Object Template** column to display the Single-Choice dialog window.

     A blank cell in this column indicates that the **Default Text Object Template** or the **Default Object Template** is currently assigned, depending on the type definition.

     . Check the checkbox for the new object template, and then click **OK** to close this dialog window.

     When data is exported with this document template, the object template that you choose overrides the one shown by the **Object Template** property for that type definition. For more information, see Modifying the Properties of a Type Definition in chapter 3, *Customizing Object Types*.

     Repeat these steps for each additional type definition whose object template you want to change.

- To display full content reference links as hyperlink URLs in the **Preview** tab and window:

  . Double-click the **Document Template Rules** property value to display the Single-Choice dialog window.

  . Check the checkbox for **Preserve Links**, and then click **OK** to close this dialog window.

  For more information about reference links, see the *Systems Architect/Requirements Management User's Manual*.

4. To save your changes in the database and close the Document Template dialog window, click **Save**.

Procedure Notes

Step 2: You can also right-click the template and choose **Open** from the popup menu. Or, double-click the template.

Object Templates

An object template contains *tags* that represent object properties, whose values are extracted to the export document for each object that the user specifies. By default, one of two object templates is assigned to each type definition, including the system-defined subtypes for folders (**Document**) and requirements (**Paragraph**).

- The **Default Text Object Template** is assigned to requirements, paragraphs, and notes. This template contains two tags. The **{%Name}** tag represents the **Name** property, whose value is an object name. The **{%HTML}** tag represents the **HTML** property, whose value is the full content of a requirement, paragraph, or note, including graphics and tables.

  The **{%Name}** tag occurs in the template heading, which is formatted in Word's Heading 1 style, and the **{%HTML}** tag occurs in the template body. With this template, each object name is extracted to a heading in the document and the full content of the object follows in the body. Heading levels in the document depend on the way in which the user specifies the objects.

  The **HTML** property is not displayed in the Administration or Systems Engineering and Requirements Management module because the value can exceed the effective size for table cells.

- The **Default Object Template** is assigned to folders, building blocks, and groups. This template contains only the **{%Name}** tag. The tag represents the **Name** property, whose value is an object name, and occurs in the template heading, formatted in Word's Heading 1 style. With this template, each object name is extracted to a heading in the document. Heading levels in the document depend on the way in which the user specifies the objects.

The project administrator can modify the default templates to customize exported data for the related object types. Also, custom object templates can be created to export different data for different object types.

The default object templates are overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up customized default templates from the earlier version before upgrading to a later version.

In any object template, tags can be added for other properties to include that data in the export document, and tags can be deleted to exclude that data. Tags can be added to run activators when the export document is generated. By adding tags for notes, note content and properties can be appended to exported data.

Boilerplate can be created by inserting standard text, special keyboard characters, graphics, symbols, equations, and hyperlinks.

For more information, see Creating an Object Template and Modifying an Object Template, later in this chapter.

Creating an Object Template

An object template determines the data that is exported to Microsoft Office Word for the type definitions to which the object template applies. Tags in the object template represent object properties, whose values are extracted to the export document for each specified object. You can customize the exported data by creating unique object templates, containing tags that you enter, and then assigning your object templates to type definitions through a document template.

Initially, a new object template contains only the **{%Name}** tag, which represents the **Name** property as in the **Default Object Template**. After you create the object template, you can add tags for any other properties that suit your purpose, and you can add tags for activators and notes. Also in the object template, you can create boilerplate for the export document. For more information, see Object Templates earlier in this chapter, and Modifying an Object Template, later in this chapter.

Before or after you modify the new object template, you can assign it to type definitions through the **Default Document Template** and through document templates that you create. For more information, see Creating a Document Template and Modifying a Document Template, earlier in this chapter.

- You must ensure there are no empty headers while creating an object template. Export using the object template generates no content if there are empty headers.
- The content before the first header in the object template is not used during export. This behavior is similar to import wherein any content before the first header becomes a note.

To create an object template:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Pull down the **File** menu and choose the **New→Object Template** options.

   The content table displays the new object template with a default name in an open text field.

3. Enter the template name, and then press the enter key.

   To refresh the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the **Reports and Formatting** folder and choose **Refresh** from the popup menu. You can also click the **Refresh** button on the toolbar.

Procedure Notes

Step 2: You can also right-click the **Reports and Formatting** folder and choose the **New→Object Template** options from the popup menu. Or, right-click in the content table and choose **New→Template→ Object Template** from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Modifying an Object Template

To customize exported data for an object type, you can modify the tags in the object template that is assigned to the type definition. The object template can be the **Default Text Object Template**, the **Default Object Template**, or an object template that you create. For more information, see Object Templates and Creating an Object Template, earlier in this chapter.

Do not modify the default templates in the Administrator module. If you need to modify a template, copy the template and make the required changes in the copied template.

The default object templates are overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up customized default templates from the earlier version before upgrading to a later version.

In any object template, you can add tags for properties that you want to include in the export document. For example, you can add the **{%ROIN}** tag to extract the Requirement Object Identification Number (ROIN) of each requirement and paragraph that the user specifies. You can also do the following:

- Add tags that run activators when the export document is generated.
- Add tags for notes to append note content and properties to exported data.
- Delete existing tags to exclude that data from the document, and copy or move tags within the template.
- Create boilerplate by inserting standard text, special keyboard characters, graphics, symbols, equations, and hyperlinks.

An object template can contain any number of tags. They can be entered anywhere in the template, in elements such as headings, body paragraphs, lists, and tables. The elements can be formatted in any available Word styles. However, headings must be formatted in Word's built-in heading styles.

To generate headings in the document, Architect/Requirements uses only the first heading style in the template. Data for all tags in that heading is exported to a heading for each object. Data for tags in all other elements is exported to the body, regardless of the formatting style.

As in the default templates, tags can be set apart in separate elements. Tags also can be entered in conjunction with boilerplate, at any location within the same element. For example, brackets (**[ ]**) can be inserted around the **{%ROIN}** tag to enclose each ROIN with brackets in the document.

However, if you have an Outline Level (Heading Style) in the object template, it is treated as the initial object. Anything inserted before the heading row in the template is ignored. If the object template has no Outline Level (Heading Style) then everything is evaluated. If you want to insert an activator so that it is executed and its output appears before the heading, put the activator tag at the beginning of the heading row. For example:

{%Activator%ConditionalPageBreak}{%NAME}

```
{%HTML}
```

After you save your changes, Architect/Requirements references the modified template for each exported object of that type. Although only one object template can be assigned to any type definition, different subtypes with different object templates can be exported to the same document.

- Microsoft Office Excel 2013, or Excel 2016 must be installed on your computer.
- Tools for checking spelling and grammar should be disabled in Word before you save the template.

To modify an object template:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Select the template, pull down the **File** menu, and choose **Open**.

   The template opens in Word as an **.mhtml** file.

3. Do any or all of the following:

   -

     To add tags:

     - For an object property, enter **{%***Property-Name***}**.
     - For an activator, enter **{%Activator%***Name***}**.
     - For a note, enter **{%Note%***Name***%***Property-Name***}**.

       The note should be attached to a requirement that is specified for export.

     Replace *Property-Name* and *Name* with the exact name displayed in the Systems Engineering and Requirements Management or Administration module. All tags are case sensitive.

     A property or note tag can contain any property name, whether user-defined or system-defined. For more information about system-defined properties, see appendix System-Defined Properties in the Administration Module and the corresponding appendix for the Systems Engineering and Requirements Management module in the *Systems Architect/Requirements Management User's Manual*.

   - To delete, move, or copy existing tags, use the related Word functions.
   - To create boilerplate, use the Word functions for inserting standard text, special characters, graphics, symbols, equations, or hyperlinks.
   - To apply character and paragraph formatting, use the manual formatting features or the list of available styles in Word.

     Formatting applied to an **{%HTML}** tag in the template has no effect in the export document. Because the referenced HTML text may contain several styles, the appearance of exported HTML is set by the formatting within the requirement, paragraph, or note content.

4. Save your changes in Microsoft Word.

Procedure Notes

Step 2: You can also right-click the template and choose **Open** from the popup menu. Or, double-click the template.

Style Sheets

A style sheet determines the Microsoft Office Word formatting that is applied to the data in the export document. Architect/Requirements creates the **Default Style Sheet** for every new project, in the **Reports and Formatting** folder. This style sheet is automatically associated with the **Default Document Template**.

The **Default Style Sheet** can be modified to change the attributes of existing styles and to create new styles. Also, custom style sheets can be created and associated with the **Default Document Template** and with custom document templates. In any style sheet, styles can be created and modified as in any other Word document. Styles can also be imported from an existing Word document. For more information, see Creating a Style Sheet and Modifying a Style Sheet, later in this chapter.

The default style sheet is overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up a customized default style sheet from the earlier version before upgrading to a later version.

The **Default Style Sheet** contains styles for Heading 1 through Heading 9. Siemens PLM Software recommends that you add those styles to style sheets from earlier Architect/Requirements versions and to custom style sheets in your current version.

Creating a Style Sheet

A style sheet determines the Microsoft Office Word styles in which data is formatted in the export document. You can specify that formatting in advance by creating a style sheet containing custom styles, and then associating your style sheet with a document template.

Initially, a new style sheet contains the same styles as the **Default Style Sheet**. After creating the style sheet, you can modify existing styles and create new styles as in any other Word document. You can also import styles from an existing Word document. For more information, see Modifying a Style Sheet, later in this chapter.

Before or after you modify the style sheet, you can associate it with the **Default Document Template** and with a custom document template, by changing the current style sheet for the document template. For more information, see Creating a Document Template and Modifying a Document Template, earlier in this chapter.

To create a style sheet:

1. With the **Reports and Formatting** folder open, pull down the **File** menu and choose the **New→Style Sheet** options.

   The content table displays the style sheet with a default name in an open text field.

2. Enter the style sheet name, and then press the enter key.

   To refresh the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the **Reports and Formatting** folder and choose **Refresh** from the popup menu. You can also click the **Refresh** button on the toolbar.

Procedure Notes

Step 1: You can also right-click the **Reports and Formatting** folder and choose the **New→Style Sheet** options from the popup menu. Or, right-click in the content table and choose **New→Template→Style Sheet** from the popup menu.

Step 2: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Modifying a Style Sheet

When you create a style sheet, it initially contains the same styles as the **Default Style Sheet**. You can modify those styles in the new style sheet, and you can create styles to further customize the formatting in the export document. In addition, you can modify and create styles in the **Default Style Sheet**.

The default style sheet is overwritten when a new Architect/Requirements version is installed. Siemens PLM Software recommends that you back up a customized default style sheet from the earlier version before upgrading to a later version.

To create and modify styles automatically, you can import styles from an existing Word document. You can also open the style sheet in Word to enter and change styles directly, as in any other Word document.

Microsoft Office Excel 2013 or Excel 2016 must be installed on your computer.

To import styles to a style sheet:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Select the style sheet, pull down the **File** menu, and choose **Import Style Sheet**.

   A new Word session starts with the Open dialog window, which lists existing folders and files in the current drive or folder. If the list does not display the document that contains the styles, use the **Look in** field to change the drive or folder, or use the **Files of type** field to display other types of files.

3. In the list, select the document that contains the styles, and then click **Open**.

   The document opens in Word, and a message states that Architect/Requirements is importing the file. You can click **OK** to close this message and perform other actions while the import is in progress. When the import is complete, the document closes and a confirmation message is displayed.

Procedure Notes

Step 2: You can also right-click the style sheet and choose **Import Style Sheet** from the popup menu.

To enter or change styles directly in a style sheet:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Select the style sheet, pull down the **File** menu, and choose **Open**.

   The style sheet opens in Word as an **.mhtml** file. Through Word's **Format** menu, you can display the **Styles and Formatting** task pane to create, modify, or delete styles.

3. To save your changes in the database, pull down Word's **File** menu and choose **Save**, or click the **Save** button on Word's toolbar.

Procedure Notes

Step 2: You can also right-click the style sheet and choose **Open** from the popup menu. Or, double-click the style sheet.

If the information required to view graphics in a browser is missing in the **Default Style Sheet**, the graphics do not appear in the **Preview** tab. This condition occurs when a project or schema is imported from TcSE version prior to TcSE 8.0. To ensure that the graphics appear in the **Preview** tab, use schema import to obtain a more recent copy of the **Default Style Sheet**.

To update the Default Style Sheet to support graphics:

1. In the administration module, select the **Reports and Formatting** folder.
2. Select the **Default Style Sheet** and open it for edit in Microsoft Word.
3. Insert a graphics file into the document.

   Select **Insert →Picture→From File**.

4. Exit Word and save.
5. Open the **Default Style Sheet** in Word again.
6. Select the graphic and delete it.
7. Exit Word and save.

Templates for Export to Microsoft Office Excel

When users export objects to Microsoft Office Excel, Architect/Requirements references an *Excel template* in extracting data from the database to an Excel spreadsheet. To customize the data that is exported, the project administrator enters *tags* that represent data values in the template's property columns. In the template's *rule table*, the project administrator enters *key fields* that filter the tag data according to specific criteria. The rule table also filters data for reports that are output to Excel from the Search module. For more information about exporting objects to Microsoft Office Excel or using the Search module, see the *Systems Architect/Requirements Management User's Manual*.

The **Default Excel Template** is created automatically in the **Reports and Formatting** folder for every new project in the Administration module. Using Excel, the project administrator can modify the **Default Excel Template** and can create and modify custom Excel templates for specialized purposes.

Creating an Excel Template Object

After you create an Excel template object, you can delete default data tags in the new template, and you can add tags for other properties and for activators. You can enter key fields in the rule table to define custom criteria for exported data. In addition, you can create boilerplate to include with the tag data. Users can choose the template in the Systems Engineering and Requirements Management module when they export objects to Excel. For more information, see Modifying an Excel Template, later in this chapter.

To create an Excel template object:

1. In the navigation tree or the content table, open the **Reports and Formatting** folder for the project.

   The content table displays the templates, style sheets, and reports in the project.

2. Pull down the **File** menu and choose the **New→Excel Template** options.

   The content table displays the new Excel template with a default name in an open text field.

3. Enter the template name, and then press the enter key.

   To refresh the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the **Reports and Formatting** folder and choose **Refresh** from the popup menu. You can also click the **Refresh** button on the toolbar.

Procedure Notes

Step 2: You can also right-click the **Reports and Formatting** folder and choose the **New→Excel Template** options from the popup menu. Or, right-click in the content table and choose **New→Template→Excel Template** from the popup menu.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Modifying an Excel Template

For an Excel template selected in the **Reports and Formatting** folder, you can open the template in Excel to enter changes directly, and you can import content from an Excel template that is stored on a local or shared drive. For more information, see Excel Template Modification Concepts, later in this chapter.

- If the template contains multiple worksheets and if the **<start>** tag and the **<end>** tag are not entered on a worksheet, all entries on that worksheet are exported as constant values.
- If a rule is duplicated on two or more worksheets, duplicate data is included in the export file.
- If the template has the **<Start>** tag in the first row, a **–1** error is displayed. Ensure that the **<Start>** tag is in the **A2** cell or below in the Excel worksheet.

To enter changes directly in an Excel template:

1. In the **Reports and Formatting** folder, select the Excel template, and then pull down the **File** menu and choose **Open**.

   You can also right-click the template and choose **Open** from the popup menu. Or, double-click the template.

2. In the **.xlsm** Excel file, do any or all of the following:

   - To add tags in property columns:

     o For an object property, enter **{%***Property-Name***}**.

     Replace *Property-Name* with the exact name. The tag is case sensitive. A property tag can name any user-defined or system-defined property. For more information about system-defined properties, see appendix System-Defined Properties in the Administration Module and appendix B in the *Systems Architect/Requirements Management User's Manual*.

     In a cell with a tag for a date property, apply an Excel **Date** format. Without a **Date** format, the exported value is the Excel numeric equivalent of the date. You can apply a **Date** format through Excel's Format Cells dialog window.

     A **Date** format is applied by default to the **Create Time** property, which is included automatically in the **Default Excel Template** and in new Excel templates that you create. Siemens PLM Software recommends that you apply a **Date** format to all date properties that you add in any Excel template, and to date properties in templates from older versions of Architect/Requirements.

     o For an activator, enter **{%Activator%***Name***}**.

     Replace *Name* with the exact name. The tag is case sensitive. For more information, see Activator Tag Results, later in this chapter.

- To modify the rule table:
  - For a key field in the **Level** column, enter **{%L-*n*}**.

    Replace *n* with the number that represents the level of the objects to which the rule applies.
  - For a key field in the **Relationship** column, enter **{%R-*Relationship-Name*}**

    Replace *Relationship-Name* with the relationship to which the rule applies. The syntax must match that of a **Relationship** field option for an **ADD** statement. For more information about **ADD** statements, see the *Systems Architect/Requirements Management User's Manual*.
  - For a key field in the **Subtype** column, enter **{%S-*Subtype-Name*}**

    Replace *Subtype-Name* with the name of the system-defined or user-defined type definition to which the rule applies.

  For more information, see Rule Table Concepts, later in this chapter.
- To delete, move, or copy existing tags or key fields, use the Excel functions.
- To create boilerplate, use the Excel functions for inserting elements such as standard text, special characters, symbols, or hyperlinks.
- To apply cell formatting, use the Excel formatting features.
- To use formulas, use the following guideline:

  A formula in a row between the **<start>** and **<end>** tags is copied to the rows in the exported document that uses the rule row. The formula text is copied as it exists in the source. Explicit row references in the formula may not work as required as they are not adjusted. For example, to add numeric property values in columns B and C, the formula `=SUM(B4,C4)` does not work. You must use `=INDIRECT("B"&ROW())+INDIRECT("C"&ROW())` instead.
- To apply cell merges, use the following guideline:

  Cell merges on a rule row are applied to exported rows that use the rule.

3. To commit your changes to the database, save the changes in Excel.

To import content to an Excel template:

1. In the **Reports and Formatting** folder, select the Excel template that you want to modify, and then pull down the **File** menu and choose **Import→ Excel Template**.

   Architect/Requirements displays the Open dialog window, which lists existing folders and files in the current drive or folder. If the list does not display the import template, use the **Look in** field to change the drive or folder.

   The import file must have **.xlsm** or **.xlsx** file name extension.

2. In the list, select the Excel template whose content you want to import, and then click **Open**.

   This action overwrites the content of the template selected in step 1.

A message states that Architect/Requirements is importing the file. You can close this message and perform other actions while the import is in progress. When the import is complete, a confirmation message is displayed.

Procedure Notes

Step 1: You can also right-click the Excel template and choose **Import→Excel Template** from the popup menu.

Packing Multiple Objects on One Row

Grouping multiple identical objects in one level of a hierarchy is known as *packing*. When the **Level** key field of the rule table is used in conjunction with the **Excel Template Rules** property of the template, data for two or more objects can be placed on the same row of Microsoft Office Excel export files.

For more information, see Modifying an Excel Template, earlier in this chapter, and Rule Table Concepts and Data Placement in Export Files, later in this chapter.

This feature applies only if the user exports all objects in the view. If the user exports selected objects only, data for each object is placed on a separate row.

To pack multiple objects on one row:

1. In the **Reports and Formatting** folder, select the Excel template.
2. In the **Properties** tab or window, double-click the value for the **Excel Template Rules** property to display the Multi-Choice dialog window.
3. Check the checkbox for **Apply Packing**.
4. In the **Level** column of the rule table, enter key fields according to the way you want to fill the rows in the export file.

Excel Template Modification Concepts

A new Excel template object initially contains the same property columns, tags, and rule table as the **Default Excel Template**. You can modify a new template to customize the data that is exported when users choose the template. Also, you can modify an existing custom template and the **Default Excel Template** itself.

To modify a template automatically, you can import content from an Excel template that is stored on a local or shared drive. You can also open a template in Excel to enter changes directly, as you do the following in any other Excel file:

- Add and remove columns for system-defined and user-defined properties that you want to include in the export spreadsheet.
- Add rows and tags for data that you want to export, delete existing rows and tags to exclude that data, and modify, copy, and move rows and tags.
- Add tags that run activators when the spreadsheet is generated.

  An activator can return either of the following:

  - The current value of a property, which may be associated with any object in the database.
  - A constant value.

  For more information, see Activator Tag Results, later in this chapter.
- Create boilerplate above the **<start>** tag and below the **<end>** tag.

  Boilerplate can include elements such as:

  - Cell merges.
  - Standard text.
  - Special characters.
  - Symbols.
  - Hyperlinks.
  - Formulas.

  Microsoft Excel shifts the rows below the **<end>** tag downwards to accommodate exported objects. You must account for the shift in the rows in the formula that you write.

  You can also enter those elements as constant values in the cells of property columns.

  Boilerplate entered in or to the right of the rule table is not exported.

- Format cells containing tags and boilerplate to apply that formatting to the data in the corresponding cells of the export spreadsheet.
- Modify the rule table to define custom criteria for exported data.

  For more information, see Rule Table Concepts, later in this chapter.

- Enter Visual Basic macros that users can run from the export file.

  If a live Excel file that contains macros is opened on a computer where the live Office interface is not installed:

  - No message is displayed to warn that live Office is not installed.
  - The file cannot be connected to the database.
  - Changes in the spreadsheet are not accumulated and cannot be applied to properties in the database.

  If you or other users plan to distribute such a file (for example, by E-mail), Siemens PLM Software recommends that recipients be notified of these conditions.

- Insert multiple worksheets to separate data in the export file.

  Each additional worksheet initially contains empty cells. In the formats described in this procedure, you enter the tags, rule table, and boilerplate that you want to include on the corresponding worksheet in the export file. You can also enter Visual Basic macros and delete worksheets.

  - If the **<start>** tag and the **<end>** tag are not entered on each worksheet, all entries on that worksheet are exported as constant values.
  - If a rule is duplicated on two or more worksheets, duplicate data is included in the export file.
  - If you have a special character, such as â€œ->â€, â€œ*â€, or â€œ@â€, in the name of a worksheet in an Excel template, the Excel workbook exported as live workbook with this template may not work as expected. This is because of the **xlsx** or **xlsm** encoding of the special characters. Static export has no such problem.

Activator Tag Results

In an activator tag, the activator named with the **{%Activator%***Name***}** syntax can return its result in either of two forms:

- A static string to be displayed in that cell of the exported Excel spreadsheet.

  The returned string appears exactly as it does in the cell. Even if the spreadsheet is exported to live Excel, this text is not associated with any object or property.

- An object and property reference to create a *live* cell.

  The activator script must be coded to return a string in the form **:***nn.n.nnnn***:***propName***:**, where *nn.n.nnnn* is the LOID of an object (in valid format) and *propName* is the name of a property on that object.

  In the Excel export file, the cell displays the current value of that property on that object. If the spreadsheet is exported to live Excel, cells exported in this form are associated with the object and property and can be edited in the same way as other live cells.

For more information about activators, see the *Systems Architect/Requirements Management API Reference* manual.

Rule Table Concepts

The rule table begins with the column that contains a **<rule>** field in each cell. Each **<rule>** field delimits a rule that filters the tag data in the property columns to the left in the same row.

Key Fields

To the right of a **<rule>** field, three key fields are used to define the criteria that comprise the rule for that row:

- In the **Level** and **Relationship** columns, the key fields on a given row map the corresponding property tag data to the Excel file as follows:
  - The **Level** key field specifies the level of the objects for which the property tag data is exported. For example, with **2** as the key field argument, data is exported for objects that appear at level two in the view.

    The level number does not imply a hierarchy but instead reflects the relationship in the **Relationship** key field. Level precedence is according to numeric value. For example, level **2** is greater than level **1**.

  - The **Relationship** key field applies when search results are output directly to Excel or are exported to Excel from the Search Results dialog window. The field specifies the relationship to which the rule applies. For example, with **Complying Objects** as the key field argument, data is exported for objects that appear in the search results because they comply with other objects.

  The **Level** and **Relationship** key fields are similar to **ADD** statements in the Advanced Search view. Therefore, you can run report scripts in the Search Results dialog window as a model to construct rules in your template. For more information about the Advanced Search view, see the *Systems Architect/Requirements Management User's Manual*.

- In the **Subtype** column, the key field names the type definition for which the property tag data is exported. For example, with **Requirement** as the key field argument, data is exported for requirements that are selected for export to Excel, and also for requirements that are included in search results.

Levels and Relationships

Levels and relationships depend on how the user specifies the objects to export:

- When only selected objects are exported from a view, the level is **1** and the relationship is **Member** for each object.
- When all visible objects are exported from a view, the level is **1** for the leftmost objects, with level numbers increasing from left to right by numeric value.

  The relationship depends on the view from which the objects are exported:

  - From the content table, the relationship is **Member**.
  - From the **Trace** subtab of the **Links** tab or window:
    - For the **Defining Trace** pane, the relationship is **Defining**.
    - For the **Complying Trace** pane, the relationship is **Complying**.
  - From the **Complying Object Traceability** window, the relationship is **Complying**.
  - From the **Defining Object Traceability** window, the relationship is **Defining**.
  - From the Search Results dialog window, the relationship is the one used in the script to populate the dialog window. This is also the relationship for search results that are output directly to Excel.

Data Placement in Export Files

When users export objects to Excel, objects in the database are examined to determine if their levels match the key fields in the rule table of the selected template. For the object that is currently being processed, data is placed in the export file as follows:

- If the level of the current object is lower than the level of the previous object, data for the current object is placed in a new row.
- If the level of the current object is greater than or equal to the level of the previous object:
  - If the cells following the previous object are empty, data for the current object is placed in the same row.
  - If the cells following the previous object contain any character, data for the current object is placed in a new row.

  - A cell that contains a space is considered to be empty. To keep a cell from being overwritten, you must enter a visible character in the cell.
  - Siemens PLM Software recommends that you do not apply any formatting to a cell where you intend to begin one object on the same row after another object. Cell formatting may be processed as content, which in turn may prevent this data placement.

For example, assume the following:

- Objects **A**, **D**, and **E** are at level one.
- Objects **B**, **C**, **F**, and **G** are at level two.

Assume also that tags for properties named **1**, **2**, **3**, and **4** are entered in the property columns of the template, with key fields entered in the rule table as in the example below:

					Level	**Relationship**	**Subtype**
{%1}	{%2}	{%3}			{%L-1}		
			{%1}	{%4}	{%L-2}		

By default, the property tags and key fields in the example above produce the following output in the export file:

A1	**A2**	**A3**		
			B1	**B4**
			C1	**C4**
D1	**D2**	**D3**		
E1	**E2**	**E3**		
			F1	**F4**
			G1	**G4**

When the **Apply Packing** value is applied to the template's **Excel Template Rules** property:

- **B1**, **B4**, **F1**, and **F4** are placed in the row above with the parent objects.
- **C1**, **C4**, **G1**, and **G4** are placed in the row immediately below.

A1	**A2**	**A3**	**B1**	**B4**
			C1	**C4**
D1	**D2**	**D3**		
E1	**E2**	**E3**	**F1**	**F4**
			G1	**G4**

For more information, see Packing Multiple Objects on One Row, earlier in this chapter.

Data is packed only if the user exports all objects in the view. If the user exports selected objects only, data for each object is placed on a separate row.

Criteria Matching

The more key fields that you enter, the more criteria that the objects must match. Thus, a rule containing three key fields specifies the most restrictive criteria, and data is extracted for fewer objects. With a rule containing only the **Subtype** key field, for example, data is extracted for all objects of the specified type definition.

Beginning with the topmost rule in the table, rules filter tag data as follows:

- If an object is found that matches all criteria in the current rule, the tag data is extracted to the spreadsheet and rule processing stops for that object. Then, rule processing for another object starts with the topmost rule.

  - If any portion of a key field is blank, that criteria is matched by default.
  - If the current rule is all blank, all criteria are matched by default. Therefore, Siemens PLM Software recommends that you place rules containing more key fields above rules with fewer criteria.

- If an object does not match any rule in the template, no tag data is extracted for that object. When the spreadsheet is generated, a warning message states that objects that do not match any template criteria are ignored.

Conditions

The rule table in a template is not exported to the spreadsheet, nor are the **<start>** and **<end>** tags exported. In addition, the following conditions apply:

- A **<rule>** field must precede the headings in the **Level**, **Relationship**, and **Subtype** columns. The sequence of the **Level**, **Relationship**, and **Subtype** columns must not be changed, and additional columns must not be inserted within the rule table.
- If any column in the rule table is missing, including the **<rule>** field column, an error message is displayed when the user attempts to generate a spreadsheet based on the template.
- The template must contain only one **<start>** tag and only one **<end>** tag. The **<start>** tag must appear to the left of the **<rule>** field in the rule table heading.
- All property columns must be located to the left of the rule table. Any cell content to the right of the rule table is not exported to the spreadsheet.
- All rows containing data tags must be located below the **<start>** tag and above the **<end>** tag. Cell content outside that area is exported as boilerplate.
- If a property column cell contains both a data tag and a constant value, the entire cell contents are exported as a constant value.

If properties in an Excel template do not exist in the database, the text string **Property does not exist** is placed in the property columns of the Excel export file. To change this text, select the template, double-click the **Error String** value in the **Properties** tab, and enter the new text in the open text field.

Date Style Support for Export to Microsoft Office Excel

Architect/Requirements uses locale-specific date styles to export data in Excel using views. If the date styles for client locale are not found, Architect/Requirements prompts the user to confirm before exporting the data in English (US) date format. Architect/Requirements OOTB supports English (US), English (UK), English (Australia), English (Canada), German, Dutch (Belgium), Dutch (Netherland), Korean, Japanese, Chinese, and Russian date formats.

Users can extend the support for these styles by adding locale-specific styles in a custom folder and specifying this folder path in the **Custom.Folder.Path** Web Application Configuration parameter.

Users can use the **Date Style Generator** utility to generate these custom date property file (**.properties** files) and copy them to the custom folder specified in the **Custom.Folder.Path** parameter. For more information about the prerequisites and the usage of the utility, see appendix *C, Date Style Generator Utility*.

Stencils for Microsoft Office Visio

This section assumes that you are familiar with Microsoft Office Visio and with XML.

An Architect/Requirements stencil object is referenced when users do the following in the Systems Engineering and Requirements Management or Search module:

- Create diagrams in the Architect/Requirements live interface with Microsoft Office Visio. For more information about using live Visio diagrams, see the *Systems Architect/Requirements Management User's Manual*.
- Choose **Microsoft Visio** as the output option for search results. For more information about exporting a report to Microsoft Office Visio, see the *Systems Architect/Requirements Management User's Manual*.

The stencil object determines the Visio shapes that represent objects in the Architect/Requirements client and the properties of the shapes in the diagram. In the Administration module, two stencil objects are created automatically in the **Reports and Formatting** folder of every new project:

- The **Default Stencil**, for live Visio diagrams. Because live Visio diagrams are synchronized with the Architect/Requirements database, they can be used to dynamically add, modify, and delete objects in the client.
- The **Default Static Tree Stencil**, for exporting search results to static Visio diagrams that are not synchronized with the database. In such a diagram, results are displayed in a multiple level tree according to qualifying relationships specified in the search criteria, for example, by using subordinate **ADD** statements with **FOR EACH** statements in the Advanced Search view. For more information about **FOR EACH** and **ADD** statements, see the *Systems Architect/Requirements Management User's Manual*.
- The **Port Type Stencil**, for creating port type live Visio diagrams. Because live Visio diagrams are synchronized with the Architect/Requirements database, they can be used to dynamically add, modify, and delete objects such as ports, connections, trace links, in the client.

The project administrator can modify the default stencil objects and can create and modify custom stencil objects. Every stencil object has two components:

- A Visio stencil file (**.vss**), containing the shapes that represent Architect/Requirements objects. Through the standard Visio functions, shapes can be edited in existing stencil files, and new stencil files with custom shapes can be created. For more information, see the Microsoft Office Visio documentation.

  Microsoft Office Visio 2013 or Visio 2016 must be installed on your computer.

- A mapping file (**.xml**), containing XML tags that map shapes in the stencil file to objects and properties in the Architect/Requirements database. Through an XML editor, or a text editor such as Microsoft Notepad or WordPad, tags can be edited in existing mapping files, and new mapping files with custom tags can be created. For more information, see Configuring a Mapping File, later in this chapter.

The default stencil file is **TeamcenterRequirements.vss** and the default mapping file is **tcr_visio_prop_map.xml**. The project administrator can modify each of these files. In addition, the project administrator can create stencil files and mapping files, and can apply those components to the default stencil objects and to custom stencil objects. For more information, see Creating a Stencil Object and Modifying a Stencil Object, later in this chapter.

New Visio 2013 Stencil and Diagram Formats Not Supported

Architect/Requirements 9.1.4 and later supports integration with Visio 2013. Visio 2013 introduces new formats for diagrams (**VSDX**) and stencils (**VSSX**), however, they are not supported by Architect/Requirements. This affects the Architect/Requirements Project Administrator in performing the following function:

When creating a Stencil Object in the Architect/Requirements database, the initial step is to create a Visio stencil file (**.vss**), which contains shapes that represent objects in Architect/Requirements. You create the stencil file in Microsoft Office Visio and add the shapes through the standard Visio functions. This same procedure is followed with Visio 2013. However, you must note the following:

. When using Visio 2013 to create a stencil file for use in creating a Stencil Object, save the file in **VSS** format and use ***.vss** extension.

. After clicking the folder button on the right side of the **Stencil File** field to display the **Open** dialog window, select a stencil that was created in **VSS** format.

  Architect/Requirements does not function correctly if you select stencil files created in **VSSX** format.

Configuring a Mapping File

A mapping file allows you to assign shapes in a stencil file to objects and properties in the database. To configure the file, you enter and change certain XML tags according to syntax described in this section.

Using an XML editor, or a text editor such as Microsoft WordPad or Notepad, you can create a new mapping file and enter new tags. Or, you can modify the tags in an existing mapping file, including the default mapping file, **tcr_visio_prop_map.xml**.

If you add a new shape to the stencil file, Siemens PLM Software recommends that you copy the **shapeMap** tag for a standard shape in the mapping file and modify that tag to suit your requirements.

In a mapping file, you can do any or all of the following:

- Map a type definition to live Visio master shapes.
- Map text properties on a shape.
- Map an Architect/Requirements type or subtype to a Visio shape based on properties.
- Map stylistic properties on a shape.
- Map a master shape in the stencil to a connection type definition in the database.
- Map a Visio master shape in the stencil to a trace link object type in the database.
- Map line style properties for connectors defined by Visio to a relationship shape in a live Visio static tree stencil.

Mapping a Type Definition to Live Visio Master Shapes

Through the **tcr_visio_prop_map.xml** file, you can map a type definition in Architect/Requirements to one or more master shapes in a live Visio stencil.

The **visioMaster** attribute for the **shapeMap** indicates the master shape that you are attempting to map.

```
<shapeMap visioMaster="Requirement">
   <typeMaps>
         <typeMap tcrType="Requirement" tcrSubType="" default="true" />
         <typeMap tcrType="Requirement" tcrSubType="Paragraph" />
   </typeMaps>
```

typeMap indicates the object type and subtype being mapped to this particular master. You can associate multiple type and subtype combinations to one master.

However, **default=”true”** can be only on one of the entries under the **<type maps>** tag. Default indicates that when the shape is dragged from the stencil to the page, an object of that type and subtype is created in the database.

- You must have at least one **shapeMap** tag per shape in the **TeamcenterRequirements.vss** file.
- You can map a single type or subtype to multiple master shapes. Then, users can choose various master shapes when creating objects of that type or subtype in a live Visio diagram.
    - Multiple master shapes are not available for member objects that are added to a live Visio diagram owner in Architect/Requirements while the diagram is open. The first master shape mapping for a given type or subtype in the XML file is used for corresponding member shapes.
    - When a live Visio diagram is created for a diagram owner in Architect/Requirements, the first master shape mapping for a given type or subtype in the XML file is used for corresponding member shapes.
    - When a property is updated in Architect/Requirements, the property is updated on the corresponding shapes in live Visio diagrams, regardless of their mapping sequences in the XML file.

Mapping Text Properties on a Shape

Map text properties on a shape by defining a tag such as the following:

```
<textPropertyMaps>
   <textPropertyMap tcrProp="ROIN" visioProp="1.Text" />
   <textPropertyMap tcrProp="Name" visioProp="Text" />
</textPropertyMaps>
```

For this shape, the **ROIN** property is mapped to text property of the first subshape, by the prefix **1.** preceding **Text**. To map to the **Text** property of the second subshape, for example, enter **2.Text**. If there is no prefix, the property is mapped to the **Text** property of the master shape and not to any of the subshapes.

Property-based Mapping of an Architect/Requirements Type or Subtype to a Visio Master Shape

Shapes in a Visio diagram can be created based on the property value of an Architect/Requirements object. This feature eliminates the need to maintain an extensive library of Architect/Requirements types or subtypes for each Visio shape.

When you create a shape in Visio, Architect/Requirements creates an object with the correct type or subtype and set the property value specified in the mapping defined in the property mapping file for the object's type or subtype and its property value. For example, you can create a single Architect/Requirements subtype called **Aircraft Wing**. This subtype can have a single choice property called **WingType** with **Delta**, **Straight**, and **Swept** as the possible values. When you create an **Aircraft Wing** object with **WingType** value as **Delta** and send it to Visio, a Visio shape corresponding to the **Delta** value is picked up. Similarly, when you drag a shape named **Straight** from Visio stencil to a diagram, Architect/Requirements creates an **Aircraft Wing** object and sets the **WingType** value to **Straight**.

- Multi-choice property value is not supported.
- Mapping date property value is not supported.
- The change in shape due to a change in the mapping file is not live. For example, when you change the property value of a Architect/Requirements object for which a different shape is mapped in the property mapping file, the shape will not change in Visio diagram.
- When you select a start object for any link in Architect/Requirements and its end object in Visio, the property mapping is not applied for the link that is created. You should create a link either in Architect/Requirements or in Visio completely.

You can map a property value of a stencil object to a Visio shape by defining a **tcrAttributeMap** tag in the **shapeMap** tag, as shown below:

```
<shapeMap visioMaster="Requirement">
   <typeMaps>
          <typeMap tcrType="Requirement" tcrSubType="" default="true" />
                 <tcrAttributeMap>
                        <tcrAttribute name="color" value="blue" />
                 </tcrAttributeMap>
   </typeMaps>
   ...
</shapeMap>
```

For this shape, the **color** property is mapped to the property value **blue**. All requirement objects or its subtypes created will have **blue** set as the property value for the **color** property.

A property of an attribute and the property's value should be mapped to an appropriate Architect/Requirements type or subtype in order to view the shape in a Visio diagram. The following are a few scenarios and the corresponding behavior of the property-based mapping.

- If an attribute and its value (any numeric value with or without decimal) are specified in the mapping file, and then you create a numeric property in Architect/Requirements, the value for this numeric property should be the same as specified in the mapping file For example, if the value of an attribute is **8.0** in the mapping file, the value in Architect/Requirements should also be **8.0** (not 8) for an object in order to view the shape in a Visio diagram.

- If a property and its value are mapped to a Visio shape and that property exists in Architect/Requirements but is not associated to an appropriate type or subtype, then the drag-drop functionality of that type or subtype in Visio is not supported. Also, if you try to create a diagram for a folder that contains objects with unassociated types or subtypes, the objects are not sent to Visio.
- If a property and its value are mapped to a Visio shape and that property exists in Architect/Requirements but is associated to an appropriate type or subtype with a false value, then the objects of such types or subtypes are not sent to Visio when you create a diagram for the parent folder. Also, when the objects are created in Visio, they remain in the Visio diagram despite the false values (values specified other than the values in the mapping file).
- Applicable to building blocks only:

  If a child of a building block has incorrect value for a specified attribute and the parent has a correct value, Visio displays only the parent building block object. When the parent object is opened with the child diagram, the child object is not displayed. Similarly, if a child building block object has a child with a correct value for the specified attribute, then opening the top level building block with the child diagram does not show its immediate child, and therefore, the child of its child.

Mapping Stylistic Properties on a Shape

Map stylistic properties on a shape by defining a tag such as the following:

```
<stylePropertyMap tcrProp="Number" visioProp="1.FillForegnd"
visioDefaultValue="5">
   <valueMap tcrValue="1" visioValue="3" />
   <valueMap tcrValue="2" visioValue="4" />
   valueMap tcrValue="3" visioValue="2" /
</stylePropertyMap>
```

Similar to a text property map, this maps properties like color or border on a shape. In the example above, the **Number** property is mapped to **FillForegnd** (color) of first subshape.

In addition to property mapping, you must provide value mapping as well. For example, **tcrValue=”1”** is mapped to **visioValue=”3”** (green), and **tcrValue=”2”** is mapped to **visioValue=”4”** (blue). If the Architect/Requirements value does not match any in the **valueMap**, **visioDefaultValue** is used.

Mapping a Master Shape to a Connection Type Definition

Map a Visio master shape in the stencil to a connection object type in the database.

The **Connection** attribute denotes that this master shape is of the **ConnectionDB** type.

```
<shapeMap visioMaster="Connection" Connection="true">
   <typeMaps>
          <typeMap tcrType="Connection" tcrSubType="" default="true" />
   </typeMaps>
   <propertyMaps>
          <textPropertyMaps>
                <textPropertyMap tcrProp="Name" visioProp="Text" />
          </textPropertyMaps>
   </propertyMaps>
</shapeMap>
```

A connection can be configured to create only a particular object type at each end of the connection. In the following example, the connection is configured to create only objects of the **IDEF0 Input Port** type at the terminating end and only objects of the **IDEF0 Output Port** type at the originating end.

```
<PortSubtypeMaps>
   <PortSubtypeMap connectionToEnd = "IDEF0 Input Port"
   connectionFromEnd = "IDEF0 Output Port"/>
</PortSubtypeMaps>
```

Mapping a Master Shape to a Port Type Definition

Map a Visio master shape in the stencil to a port type definition in the database.

```
<shapeMap visioMaster=”Port”>
   <typemaps>
          <typeMap tcrType=”Port” direction = “External” tcrSubType=""
          default="true" />
   </typemaps>
```

The “External” value allows the port to be attached to a building block or a building block subtype.

Mapping a Master Shape to a Port Subtype

Map a Visio master shape in the stencil to a port subtype in the database.

- The stencil object's **Uses Ports** property value must be set to **Yes**.
- Each external port subtype master shape must have a corresponding internal port subtype master shape.

```
<shapeMap visioMaster="IDEF0 Input Port">
   <typeMaps>
         <typeMap tcrType="Port" direction = "External"
         tcrSubType="IDEF0 Input Port" default="true" />
   </typeMaps>
```

The subtype must exist in the **Type Definitions** folder. Type the exact name of the subtype as the **tcrSubType** attribute value. For more information, see Creating a Subtype in chapter 3, *Customizing Object Types*.

You can use the following attributes for additional control of the port location and the direction of the connection that the port can accept:

Attribute	Value	
ConnectBBposition	**Left**	Port attaches to the left side of the building block.
	Right	Port attaches to the right side of the building block.
	Top	Port attaches to the top of the building block.
	Bottom	Port attaches to the bottom of the building block.
portDir	**In**	Port accepts incoming connection.
	Out	Port accepts outgoing connection.

For example:

```
<shapeMap visioMaster="IDEF0 Input Port">
   <typeMaps>
         <typeMap tcrType="Port" direction = "External"
         tcrSubType="IDEF0 Input Port" ConnectBBposition=”Left”
         portDir=”In” default="true" />
   </typeMaps>
```

Mapping a Master Shape to a Trace Link Type Definition

Map a Visio master shape in the stencil to the trace link object type in the database.

The **Trace Link** attribute denotes that this master shape is of the **Tracelink** type.

```
<shapeMap visioMaster="Trace Link" Tracelink="true">
   <typeMaps>
          <typeMap tcrType="Trace Link" tcrSubType="" default="true" />
   </typeMaps>
   <propertyMaps>
          <textPropertyMaps>
                <textPropertyMap tcrProp="Name" visioProp="Text" />
          </textPropertyMaps>
   </propertyMaps>
</shapeMap>
```

Mapping Line Style Properties for Connections Defined by Visio to a Relationship Shape in a Live Visio Static Tree Stencil

Map line style properties for connectors defined by Visio to a relationship shape in a live Visio static tree stencil.

The **tcr_visio_prop_map.xml** file contains one section for each relationship on which the user can run a search in the Architect/Requirements client. The following example represents one of the sections in the property mapping XML file for a live Visio static tree diagram stencil:

```
<relationshipMap type="Defining Objects">
   <Line Pattern="1" Weight="0.12" Cap="0" Transparency="0">
          <Color Numeric="2" Red="" Green="" Blue="" />
   </Line>
   <LineEnds Begin="0" End="0" BeginSize="Medium" EndSize="Medium" />
   <RoundCorners Rounding="0" />
</relationshipMap>
```

To set the values in the XML section, adjust the desired values in the Line dialog window. Open the Line dialog window for the connector that is used for showing the relationships in a static tree diagram. The **Line** subsection of the XML file represents the **Line** section of the dialog window. Similarly, the **LineEnds** and **RoundCorners** sections in the XML file represent the corresponding sections of the dialog window.

The **Line** section contains a **Color** subsection that corresponds to the **Color** field in the dialog window. Each can also have custom colors with different combinations of red, green, and blue (RGB), as well as Visio defined colors represented by numeric codes. Thus, the **Color** subsection has attributes for setting numeric values for color as well as for setting custom RGB color on a relationship shape.

To get the Visio values that represent a particular style, select the relationship shape, and then pull down the **Window** menu and choose the **Shape Sheet→Line Format** options to display the **Line Format** section. Then, copy the numeric values from the **Line Format** section and add the values to the XML file for the corresponding relationship section.

For **LineColor**, copy only the values in the parentheses of the formula. Do not copy the entire formula, for example, **HSL (***n*, *n*, *n***)**.

Creating a Stencil Object

Every stencil object consists of two component files. For a new stencil object, you create the following files on your computer:

- A stencil file (**.vss**), which contains shapes that represent objects in Architect/Requirements. You create this file in Microsoft Office Visio and add the shapes through the standard Visio functions. For more information, see the Microsoft Office Visio documentation.

  Microsoft Office Visio 2013 or Visio 2016 must be installed on your computer.

- A mapping file (**.xml**), which contains XML tags that map shapes in the stencil file to objects and properties in the Architect/Requirements database. You use an XML editor, such as Microsoft WordPad or Notepad, to create this file and add the tags. For more information, see Configuring a Mapping File, earlier in this chapter.

After you create the custom components, you create the new stencil object by loading the components from your computer to Architect/Requirements.

To create a stencil object:

1. On your computer, create the stencil file and the mapping file that you want to associate with the new stencil object.
2. Open the **Reports and Formatting** folder, and then pull down the **File** menu and choose **New→Template→Stencil**.

   The Select dialog window is displayed.
3. Associate your custom components with the stencil object by doing the following:
   a. Click the folder button to the right of the **Stencil File** field to display the Open dialog window.
   b. Select the stencil file, and then click **Open** to fill in the field automatically.
   c. Click the folder button to the right of the **Mapping XML File** field to display the Open dialog window.
   d. Select the mapping file, and then click **Open** to fill in the field automatically.
4. To load the components and create the stencil object, click **OK**.

   The content table displays the new stencil object with a default name in an open text field.
5. Enter the stencil object name, and then press the enter key.

   To refresh the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the **Reports and Formatting** folder and choose **Refresh** from the popup menu. You can also click the **Refresh** button on the toolbar.

Procedure Notes

Step 3: In the Open dialog window, you can use the **Look in** field to change the drive or folder.

Modifying a Stencil Object

By modifying stencil objects, you can customize the shapes in live Visio diagrams and the shapes in search results that users output to Microsoft Office Visio. Two default stencil objects are generated automatically for every new project, in the **Reports and Formatting** folder. The folder also contains stencil objects that you create. For more information, see Creating a Stencil Object, earlier in this chapter.

Every stencil object consists of two component files:

- A stencil file (**.vss**), where you can add and change shapes through the standard Visio functions. The default stencil file is **Default Stencil.vss**.

  Microsoft Office Visio 2013 or Visio 2016 must be installed on your computer.

- A mapping file (**.xml**), where you can add and change tags that define the properties of shapes in the stencil file. The default mapping file is **tcr_visio_prop_map.xml**. You edit a mapping file with a text editor, such as Microsoft Notepad or WordPad, or an XML editor. For more information, see Configuring a Mapping File, earlier in this chapter.

You can modify any stencil object, including the **Default Stencil** (for live Visio diagrams) and the **Default Static Tree Stencil** (for search results). For the selected stencil object, you copy the stencil file and the mapping file from the Architect/Requirements client to your computer. There, you edit one or both of the files. When you complete those changes, you reload each changed file from your computer to the stencil object in the client.

To modify a stencil object:

1. Copy the stencil file and the mapping file to your computer by doing the following:
   a. In the **Reports and Formatting** folder, right-click the stencil object and choose **Copy to client** from the popup menu.

      The Select Location to Copy dialog window displays the folders and files in the current drive or folder. You can use the **Look in** field to change the drive or folder.

   b.

      Select the drive or folder, and then click **Open**.

      A message confirms the location of the copied files.
2. On your computer, edit one or both of the copied files.
3.

   Pull down the **File** menu and choose **Import→Stencil** to display the Select dialog window.
4. Do one or both of the following:
   a.

      If you edited the stencil file:

      - Click the folder button to the right of the **Stencil File** field.

        The Open dialog window displays the folders and files in the current drive or folder. You can use the **Look in** field to change the drive or folder.
      - Select the edited stencil file, and then click **Open** to fill in the field automatically.

b.

If you edited the mapping file:

- Click the folder button to the right of the **Mapping XML File** field.

  The Open dialog window displays the folders and files in the current drive or folder. You can use the **Look in** field to change the drive or folder.

- Select the edited mapping file, and then click **Open** to fill in the field automatically.

If you edited only one of these files, clear the checkbox for the unchanged file.

5. To reload the edited files into the stencil object, click **OK**.

   The Select dialog window closes, and the modifications are applied to the stencil object.

If the stencil object's **Uses Port** property value was **Yes** before the modifications, ensure that you reset this value to **Yes** after reloading edited files into the stencil object.

Procedure Notes

Step 3: You can also right-click the stencil object and choose **Import→Stencil** from the popup menu. Or, double-click the stencil object.

Step 4: You can also enter the path, the file name, and the appropriate file name extension directly in each field.

Viewing Diagram Links

Architect/Requirements allows you to view the diagram links in a project. However, you must enable the display of diagram links before you can view them.

Enabling diagram links display

Diagram links are not visible by default. You must configure Architect/Requirements to view the diagram links. The configuration to view the diagram links is done on the project level.

To make the diagram links visible:

1. Select the project for which you want to view the diagram links in the **Navigation** tree.
2. Click **Properties** tab in the **Notebook** pane.
3. Scroll down and select **Project Settings**.
4. In the Project Settings window, select **Show Diagram Links** check box.
5. Click **OK**.

Viewing diagram links

You can view the diagram links in **Relations** tab of link view.

To view the diagram links:

1. Select the diagram in the **Content** table.
2. Click **Links** in the **Notebook** pane to make the **Links** tab active.
3. Open the tab in a floating window.

   Click the **Open tab in a new window** button to open the tab in a floating window.
4. Click the **Relations** tab in the **Links** window.
5. Select the diagram in the **Attachment** tab.

   The diagram links are displayed in the **Links** window.

Deleting diagram links

You can delete the diagram links if the following conditions are true:

- If Diagram Content property of a diagram is set to static.
- The owner of a diagram is in non frozen state.

To set the Diagram Content property to static:

- Select the diagram in the **Content** table.
- Right click the diagram in the **Attachment** tab and select **Properties**.
- Double click the values box for **Diagram Content**.
- In the **Diagram Content** dialog, select **Static** and click **OK**.
- Click **Close**.

To delete a diagram link:

- Set the **Diagram Content** property to static and open the diagram link in the floating **Links** window.

  For information on opening the diagram link in a floating window, see Viewing diagram links.
- Right click the diagram link that you want to delete.
- To confirm the deletion, click **Yes**.

Searching for diagram links

You can search for requirements that contain diagram links or all the diagram links in a project. You need to use the **Search** module to perform the search.

To search for diagram links:

1. Select the project in which you want to search for the diagram links and click the **Search** button in the **Module** bar.
2. Select **Requirement** in the **Types/Subtypes** list.
3. Select **Advanced** for the **Search Type**.
4. In the **Modify** group, click the **Add Indented** button.
5. In the **Query Edit** area, select **FOR EACH** for the **Command**.
6. In the **Modify** group, click the **Add Indented** button.
7. In the **Query Edit** area, select **Diagram Links** from the **Relationship** drop down list.

   The query in the **Query View** is `SELECT Requirement FOR EACH ADD Diagram Links`.
8. Click the **Search** button.

   Architect/Requirements displays the search results in a new window.

To search for all the diagram links in a diagram:

1. Select the project in which you want to search for the diagram links and click the **Search** button in the **Module** bar.
2. Select **Diagram** in the **Types/Subtypes** list.
3. Select **Advanced** for the **Search Type**.
4. In the **Modify** group, click the **Add Indented** button.
5. In the **Query Edit** area, select **FOR EACH** for the **Command**.
6. In the **Modify** group, click the **Add Indented** button.
7. In the **Query Edit** area, select **Diagram Member Links** from the **Relationship** drop down list.

   The query in the **Query View** is `SELECT Diagram FOR EACH ADD Diagram Member Links`.
8. Click the **Search** button.

   Architect/Requirements displays all the all the diagram links for the diagrams in a new window.

Guidelines for Working In Live Visio Stencils

This section describes some general guidelines for working in live Visio stencils.

Considerations for Creating Port Master Shapes

In creating port master shapes, some special considerations are involved:

- Create a port master shape as a group shape with two subshapes.

  The live Visio **Hide/Unhide** feature requires two sub-shapes. Use one subshape for displaying the port and one subshape for displaying the port name. You can add text on the port name subshape.

-

  Open the shape sheet for each subshape and add a layer membership section.

  - Add a **Port** layer for the port shape subshape.
  - Add a **PortName** layer for the port name subshape.

Connection Point Sections for Port Master Shapes

After you create a port master shape and before you save it, ensure that it has a connection point section:

- Select the port master shape, and then pull down the Visio **Window** menu and choose **Show ShapeSheet**.

  The connection point section is displayed if it exists.

  To create a connection point section:

  . Select the shape, and then pull down the Visio **Insert** menu and choose **Section**.

  The Insert Section dialog window is displayed.

  . Check the **Connection points** check box and click **OK**.

  The connection point section is added to the shape sheet.

  The connection point section initially contains one row for one connection point. You can assign the X and Y coordinates for that connection point.

  You can also insert rows for additional connection points. To rename a connection point, select the default row name and enter the new name.

Then do the following in the connection point section:

1. Delete all connection point rows from the connection point section.
2. Insert a new row in the connection point section.
3. Assign the X and Y coordinates so that the connection point is created at the center of the port shape.
4. Select the **Type/C** cell of the new row, and then select **2 – visCnnctTypeInwardOutward** from the list.
5. Ensure that only one connection point exists on the port master shape.

   Live Visio makes additional connection points available when a corresponding port shape is used on the Visio page.

6. Close the shape sheet and save the master.
7. In a new or existing mapping file, assign the port master shape to objects and properties in the Architect/Requirements database.

   For more information, see Configuring a Mapping File, earlier in this chapter.

For information about working with connection points, see the Microsoft documentation at the following URL:

http://office.microsoft.com/en-us/visio/HP010473091033.aspx

Using connection points

Connections are attached to Architect/Requirements building blocks and ports. You must specify connection points when master shapes for building blocks and ports are defined. The connection points indicate points where connections are attached. There are restrictions on where you can define the connection points for Architect/Requirements master shapes. In Microsoft Visio, master shapes are defined by grouping together multiple sub-shapes. Architect/Requirements interacts correctly only with connection points defined at the group level. If the shape sheet for a sub-shape has a **Connection Points** section, you must remove it. You can add the required connection points at the group level.

Chapter 5: Managing Users

This chapter contains an overview of users and instructions for creating new users and user groups and for managing project access, user profiles, and user passwords.

Overview of Users

Individuals are represented in Architect/Requirements by *user* objects, whose registration determines project access and licensing. When a new user is created, the user object is registered in the database and is added to the **Users** folder of the **TcSE Administration** project, through which all user related transactions are recorded.

In the **TcSE Administration** project, the **Users** folder contains all activated and deactivated user objects for all projects in the Architect/Requirements installation. In every other project, the **Users** folder contains a user object for each user who currently has access to that project. A user can be managed from any project in which the **Users** folder contains that user object. Users can be managed also through *user groups*, which consist of shortcuts to existing user objects in a project.

Project Access

For each user, a *maximum privilege level* limits the *access privilege* that can be granted in any project. In turn, the access privilege controls the user's actions in a specific project. The maximum privilege level assigned to the user determines project access as follows, from the highest maximum privilege level to the lowest:

- **Enterprise Administrator** allows the user to create new projects, to delete any existing project, and to perform any action in all projects. The user does not need project specific access privileges because enterprise administrators automatically have full access to every project.

  When an Architect/Requirements database is created through the normal installation process, the creator becomes the original enterprise administrator by default. A user object is generated for the database creator, with a user name of **tcradm** and a blank password. The maximum privilege level for that user object cannot be changed by any user, including the database creator.

- **Project Administrator** allows the user to perform any action in and to delete specific projects, those where **Project Administrator** access privilege is granted. However, the user cannot create new projects.

Power User is a non-administrative **Read and Write** user who is given certain project-specific privileges to assist with project administration. The project-specific privileges are managed through special user groups and security profiles.

- **Read and Write** allows the user to view, modify, create, and delete objects in the Systems Engineering module, and to view objects in the Administration module, in projects where **Read and Write** access privilege is granted.
- **Read Only** allows the user only to view objects in the Systems Engineering module, in projects where **Read Only** access privilege is granted. The user cannot view schema objects in the Administration module.
- **No Access** revokes access to all projects, thereby deactivating the user. The user object remains in the **Users** folder of the **TcSE Administration** project.

Except for **No Access**, a given maximum privilege level can be assigned to any number of new and existing users. **No Access** can be assigned only to existing users.

Whereas a user's maximum privilege level applies to all projects, access privileges can differ from one project to another. For example, a user may have **Project Administrator** access privilege in one project, **Read and Write** access privilege in a second project, and **Read Only** access privilege in a third project. The access privilege in any project must be equal to or lower than the current maximum privilege level. For a higher access privilege, a higher maximum privilege level must first be assigned. For more information, see Modifying a User Profile, later in this chapter.

Notwithstanding the access privilege, a user's actions in a project may be further controlled by a security profile. For more information, see Security Profiles and Access Control in chapter **Unsatisfied xref reference**, ***Unsatisfied xref title***.

Special Privileges

In addition to a user's access privilege, special privileges can be granted in specific projects:

- **Script Authoring** privilege allows a user to create, edit, and run activators in the Administration module. This privilege is granted for a project by adding **Script Authoring** to the **Additional Privilege** property of the user object. For more information about activators, see the *Systems Architect/Requirements Management API Reference* manual.
- **Architect** privilege allows a user to work directly with building blocks, TRAMs, and diagrams in the Systems Engineering module. The user can do the following:
    - Create building blocks and TRAMs, using system-defined and user-defined subtypes.
    - Promote building blocks and TRAMs to higher levels in a hierarchy.
    - Demote building blocks and TRAMs to lower levels in a hierarchy.
    - Move building blocks and TRAMs up or down within a level in a hierarchy.

- Create and edit diagrams with the Architect/Requirements live interface to Microsoft Office Visio.
- Embed diagrams in Microsoft Word documents.
- Copy building blocks, TRAMs, and diagrams to create new objects.
- Move building blocks, TRAMs, and diagrams to different owners.
- Rename building blocks and TRAMs.
- Create shortcuts to building blocks, TRAMs, and diagrams.
- Import building blocks and TRAMs from Microsoft Excel and XML files to new and existing projects.
- Delete building blocks, TRAMs, and diagrams.
- Change the values of editable properties for building blocks and TRAMs.
- Calculate numeric properties for all object types.
- Freeze and unfreeze building blocks and TRAMs.
- Create versions, variants, and baselines of building blocks and TRAMs.
- Submit changed building blocks and TRAMs for approval.

This privilege is granted for a project by adding **Architect** to the **Additional Privilege** property of the user object. Users who do not have this privilege can create notes and trace links for building blocks, TRAMs, and diagrams, and also can view those objects.

Power Users

Power users are non-administrative users who are assigned with project-specific privileges (between **Read and Write** and **Project Administrator** access privileges) to assist with the project administration. Power users perform relatively routine tasks based on the project-specific privileges assigned to them. Project administrators can add the **Power Users** package to a project and assign privileges by adding users to one of the power user groups.

To enable the **Power Users** package for a project:

1. From the **Administration** module, select the **Projects** node.

   The content table displays all projects to which you have access.

2. Select the project in the content table.

   The **Properties** tab or window displays all viewable properties for the project.

3.

   In the **Value** column, double-click the **Packages** property value.

   The Multi-Choice dialog window is displayed.

4. Check the **Power Users** checkbox, and then click **OK** to close the dialog window.

   This action cannot be reversed. The **Power Users** package cannot be removed from the project.

The **Power Users** value is added to the **Packages** property, indicating that the power user groups are enabled for the project. The power user groups can be accessed from the **Users→Power Users** folder of the project.

Power user access to administration objects is controlled by security profiles. Without a security profile, a power user has the same rights to an administration object as a read/write user. Installing the Power User package creates the security profiles and user groups for an access system based on the administration folders. The new security profiles are placed in a sub folder called **Power_User_Security_Profiles**. The user groups are placed in a sub folder called Power Users. The security profile can be easily modified or replaced when required as it uses the normal security profile mechanism. Inheritable security profiles are set on the administration folders so that new administration objects have the appropriate security profile. Out of the box objects and other administration objects created before the Power User package is enabled, do not inherit the security profile. Security profiles must be set manually on such objects if power user access is required. If power user access to certain objects is not required, the security profile can be removed. If power user access is not required for a particular administration folder, the folder's security profile can be removed or the security profile can be changed to ensure that the security profile is not applied to new descendants.

Users can be granted project-specific power user privileges by adding them to one the power user groups described below. These privileges are controlled through security profiles. To add users to a power user group, copy the selected users to the desired power user group. For more information, see Modifying a User Group, later in this chapter.

- **PU_Activators_Access** privilege allows a user to promote menu folders and items. The user can create and modify all types of activators, which also requires script authoring privilege.
- **PU_Property_Definitions_Access** privilege allows a user to create definitions for choice, numeric, text, and date properties.
- **PU_Reports_and_Formatting_Access** privilege allows a user to create documents, excel files, object templates, stencils, and style sheets. The user can also promote reports and views.
- **PU_Security_Profiles_Access** privilege allows a user to create security profiles.
- **PU_Type_Definitions_Access** privilege allows a user to create subtypes of type definitions. Before granting this privilege, project administrators must add a security profile to a type definition that grants complete access to the power users.
- **PU_User_Access** privilege allows a user to create a new user group and assign users to that group. It also allows a user to create new users in the project, and manage user properties such as name, e-mail ID, and password.

For Excel imports, a power user must be added to the following user groups:

- **PU_Type_Definitions_Access**
- **PU_Property_Definitions_Access**

Licensing

Licenses for an Architect/Requirements installation are consumed only by activated users, those who currently have access to at least one project. For such a user, licensing is determined as follows:

- A **Requirements** license is consumed when the maximum privilege level is **Enterprise Administrator**, **Project Administrator**, **Power User**, or **Read and Write**. If the user has access to two or more projects, only one license is consumed.

  For a user with this license, additional licenses can be assigned for special privileges:

  - A **Script Authoring** license is consumed, in conjunction with the **Requirements** license, when **Script Authoring** privilege is granted to the user in at least one project. If the user has this privilege in two or more projects, only one **Script Authoring** license is consumed. For more information about activators, see the *Systems Architect/Requirements Management API Reference* manual.
  - An **Architect** license is consumed, in conjunction with the **Requirements** license, when **Architect** privilege is granted to the user in at least one project. If the user has this privilege in two or more projects, only one **Architect** license is consumed.

  Additional licenses are assigned through the **Additional Privilege** property of the user object. For more information, see Special Privileges, earlier in this chapter.

- A **Consumer** license is consumed when the maximum privilege level is **Read Only**. If a **Consumer** license is not available, a **Requirements** license is consumed. If the user has access to two or more projects, only one license is consumed.

A user does not consume a license of any type when the maximum privilege level is **No Access**. The user is deactivated, although the user object remains in the **Users** folder of the **TcSE Administration** project. Users who currently have project access can be deactivated to free those licenses, which can then be assigned to other users or held in reserve. For more information, see Deactivating a User, later in this chapter.

Without deactivating a user, the maximum privilege level can be changed to free the current **Requirements** or **Consumer** license and assign the other type, if available. Also, a user's **Script Authoring** license or **Architect** license can be freed by removing the corresponding value from the **Additional Privilege** property in each project where the user has that privilege. For more information, see Modifying a User Profile, later in this chapter.

Table 1 identifies the menu options available in the Systems Engineering and Requirements Management module for each license type.

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
File	**New→Requirement**	□	□		control-R
File	**New→Paragraph**	□	□		control-H
File	**New→Building Block**		□		control-B
File	**New→Folder**	□	□		control-L
File	**New→Note**	□	□		control-N
File	**New→Group**	□	□		control-G
File	**New→TRAM**		□		
File	**New→Document**	□			
File	**New→Spreadsheet**	□	□		
File	**New→Child**	□[1]	□		control-K
File	**New→Subtype**	□[2]	□		control-U
File	**Open**	□	□	□[3]	control-O
File	**Open Read-only**	□	□	□	
File	**Excel Open**	□	□		
File	**Visio→Create Diagram**		□		
File	**Visio→Open**	□[4]	□	□[5]	
File	**Delete**	□[6]	□		delete
File	**Rename**	□[7]	□		F2

[1] Disabled if a building block is selected.
[2] Disabled for building block and TRAM subtypes.
[3] Opens object in read-only mode.
[4] Opens object in read-only mode.
[5] Opens object in read-only mode.
[6] Disabled if a building block, TRAM, or diagram is selected. Disabled also if a building block, TRAM, or diagram is subordinate to a selected object.
[7] Disabled if a building block or TRAM is selected.

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
File	**Properties**	□[8]	□		
File	**Export→Word Document**	□	□	□[9]	
File	**Export→Excel Spreadsheet**	□	□	□[10]	
File	**Export→AP233**	□	□		
File	**Export→XML**	□	□		
File	**Import→Word Document**	□	□		
File	**Import→Excel Spreadsheet**	□[11]	□		
File	**Import→AP233**	□	□		
File	**Import→XML**	□[12]	□		
File	**Exit**	□	□	□	control-Q
Edit	**Undo**	□	□		control-Z
Edit	**Cut**	□[13]	□		control-X
Edit	**Copy**	□[14]	□		control-C
Edit	**Paste**	□	□		control-V
Edit	**Paste Shortcut**	□[15]	□		
Edit	**Copy URL**	□	□		

[8] Edit Properties dialog window is read-only if a building block or TRAM is selected.
[9] Document in read-only mode.
[10] Spreadsheet in read-only mode.
[11] Creating building blocks and TRAMs is disabled.
[12] Creating building blocks and TRAMs is disabled.
[13] Disabled if a building block, TRAM, or diagram is selected.
[14] Disabled if a building block, TRAM, or diagram is selected. Disabled also if a building block, TRAM, or diagram is subordinate to a selected object.
[15] Disabled if a building block, TRAM, or diagram is selected.

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
Edit	**Copy SnapShot**		□		
Edit	**Links→Start Trace Link**	□	□		
Edit	**Links→End Trace Link**	□	□		
Edit	**Links→End Trace Link Subtype**	□	□		
Edit	**Links→Link to Application**	□	□		
Edit	**Links→Link to TcEngineering**	□	□		
Edit	**Links→Link to TcEnterprise**	□	□		
Edit	**Promote**	□ [16]	□		control-P
Edit	**Demote**	□ [17]	□		control-D
Edit	**Move Up**	□ [18]	□		
Edit	**Move Down**	□ [19]	□		
Edit	**Proxy Info**	□	□	□	
View	**Go To→Search**	□	□	□	
View	**Go To→TcSE**	□	□	□	
View	**Go To→Administration**	□	□	□	
View	**Go To→Navigation Tree**	□	□	□	
View	**Go To→Content Table**	□	□	□	
View	**Go To→Notebook Pane**	□	□	□	□

[16] Disabled if a building block is selected.
[17] Disabled if a building block is selected.
[18] Disabled if a building block is selected.
[19] Disabled if a building block is selected.

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
View	Go To→To Object	□	□	□	
View	Save	□	□	□	
View	Save As	□	□	□	
View	Set View As Default	□	□	□	
View	Module Bar	□	□	□	
View	Address Bar	□	□	□	
View	Content Table	□	□	□	
View	Navigation Tree	□	□	□	
View	Notebook Pane	□	□	□	
View	Root Node	□	□	□	
View	Format Columns	□	□	□	
View	Filter Properties→All	□	□	□	
View	Filter Properties→Editable	□	□	□	
View	Filter Properties→Read-only	□	□	□	
View	Filter Properties→System	□	□	□	
View	Filter Properties→User Defined	□	□	□	
View	Filter Properties→View Specific	□	□	□	
View	Filter Properties→Toggle Controls	□	□	□	
View	Lines Per Row→1 Line	□	□	□	

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
View	Lines Per Row→5 Lines	□	□	□	
View	Lines Per Row→10 Lines	□	□	□	
View	Lines Per Row→20 Lines	□	□	□	
View	Expand All	□	□	□	
View	Refresh	□	□	□	
Tools	Send Email	□	□		
Tools	Change Password	□	□		
Tools	Change User Info	□	□		
Tools	Calculate Properties		□		
Tools	Traceability→Complying	□	□		control-T
Tools	Traceability→Defining	□	□		
Tools	Versions→Create Variant	âœ“[20]	□		control-I
Tools	Versions→Create Version	âœ“[21]	□		control-E
Tools	Versions→Submit For Approval	âœ“[22]	□		
Tools	Versions→Freeze→ Selected Objects	âœ“[23]	□		
Tools	Versions→Freeze→ Deep	âœ“[24]	□		

[20] Disabled if a building block is selected.
[21] Disabled if a building block is selected.
[22] Disabled if a building block is selected.
[23] Disabled if a building block is selected.
[24] Disabled if a building block is selected.

Table 5-1. Menu Options for Architect/Requirements License Types: Systems Engineering Module

Menu	Option	Requirements License	Architect License	Consumer License	Shortcut Keys
Tools	**Versions→Unfreeze→ Selected Objects**	âœ“[25]	□		
Tools	**Versions→Unfreeze→ Deep**	âœ“[26]	□		
Tools	**Versions→Create Baseline**	âœ“[27]	□		
Tools	**System Information**	□	□	□	
Tools	**Search**	□	□		control-F
Tools	**Run Macro**	□	□	□	control-M
Tools	**Run Report**	□	□	□	
Help	**Manuals**	□	□	□	F1
Help	**About**	□	□	□	

[25] Disabled if a building block is selected.
[26] Disabled if a building block is selected.
[27] Disabled if a building block is selected.

Creating a New User

You can create a user in any project where you have **Project Administrator** privilege, including the **TcSE Administration** project. If you create a user in the **TcSE Administration** project, the user is added to the **Users** folder and project specific access can then be granted. When you create a user in any other project, the user automatically has access to that project and is added to the **Users** folder of the **TcSE Administration** project. For more information, see Project Access, earlier in this chapter, and Granting or Revoking Project Access, later in this chapter.

To create a new user:

1. In the navigation tree or the content table, open the **Users** folder for the project.
2. Pull down the **File** menu and choose the **New→User** options to display the Create User dialog window.
3. Enter the Architect/Requirements user name in the **User Name** field.

   You can enter the user's actual names in the **First name** and **Last name** fields.

   In the **Email** field, you can enter the user's E-mail address to enable the user to send and receive E-mail from the Architect/Requirements client. This field must be filled in to enable the function for the user. For more information about sending E-mail from the client, see the *Systems Architect/Requirements Management User's Manual*.
4. Enter the password in the **Password** and **Re-enter Password** fields.
5. In the **Privilege** field, do one of the following to assign the maximum privilege level:
   - Select **Read Only** to allow the user only to view objects in both the Systems Engineering and Requirements Management and Administration module for all projects.
   - Select **Read and Write** to allow the user to view, modify, create, and delete objects in the Systems Engineering and Requirements Management module, and to view objects in the Administration module, for all projects.
   - Select **Project Administrator** to allow the user to create new projects, to delete any existing project, and to perform any action in specific projects.
   - Select **Enterprise Administrator** to allow the user to create new projects, to delete any existing project, and to perform any action in all projects.

     With this maximum privilege level, the user is added to the **Users** folder of every project in the Architect/Requirements installation. You must have **Enterprise Administrator** privilege to assign this maximum privilege level.
6. Click **OK** to close the dialog window and display the user in the content table.

Procedure Notes

Step 2: You can also right-click the **Users** folder in the navigation tree, or right-click anywhere in the content table, and then choose the **New→User** options from the popup menu. Or, click the **New User** button on the toolbar or press control-R.

Granting or Revoking Project Access

When you create a new user in a project, the user automatically has access to that project through the maximum privilege level that you assign. You can grant access to any or all other projects for which you have **Project Administrator** privilege.

For a new or existing user, you can grant or revoke access to one project at a time. You can also grant and revoke access to multiple projects in a single action.

The user is added to the **Users** folder in each project to which access is granted. When access is revoked, the user object is removed from that **Users** folder. The user object remains in the **TcSE Administration** project when access to all projects is revoked.

To grant or revoke project access:

1. In any project that contains the user object, open the **Users** folder, and then select the user in the content table.

   The properties table displays all viewable properties that apply to the user.

   For this project only, you can revoke access and remove the user object by pulling down **File** menu and choosing **Delete**. You can also right-click the user and choose **Delete** from the popup menu. Or, click the **Delete** button on the toolbar or press the delete key.

2. In the **Values** column, double-click the **Projects** property value.

   The Multi-Choice dialog window is displayed, listing all projects for which you have **Project Administrator** privilege. Checkmarks indicate the projects to which the user currently has access.

3. Do one or both of the following:

   - Check the checkbox for each project to which you want to grant access, or click **Select All** to check all checkboxes simultaneously.
   - Clear the checkbox for each project for which you want to revoke access, or click **Unselect All** to clear all checkboxes simultaneously.

   You can check or clear checkboxes in any combination, and you can use the checkboxes and buttons together.

4. Click **OK** to apply the changes in the database and close the dialog window.

   If the user remains in this project, the **Projects** value shows the name of each project where the user now has access. If you revoked access to this project, you can see the **Projects** value in any other project where the user has access. If you revoked access to all projects, the **Projects** value is blank in the **TcSE Administration** project.

Procedure Notes

Step 1: You can also select the user in the **Users** folder of the **TcSE Administration** project. However, you cannot delete the user from that project.

Step 4: To reverse this action, you can pull down the **Edit** menu and choose **Undo Change Properties**, click the **Undo** button on the toolbar, or press control-Z.

Modifying a User Profile

A user profile consists of properties that record information about a user object. You modify a user profile by changing any of the following property values:

- **Maximum Privilege** specifies the highest access privilege that can be granted to the user for any project. The maximum privilege level applies to all projects, whether or not the user currently has access to any project. For more information, see Project Access, earlier in this chapter.
- **User Access** specifies the access privilege that is granted to the user for this project. The access privilege for any project must be equal to or lower than the maximum privilege level. A different access privilege can be granted for each project to which the user has access. For more information, see Project Access, earlier in this chapter.
- **Additional Privilege** lists special privileges that are granted to the user for this project, such as:
  - **Script Authoring** privilege allows the user to create, edit, and run activators in the Administration module.
  - **Architect** privilege allows the user to work with building blocks, TRAMs, and diagrams in the Systems Engineering and Requirements Management module.

  For more information, see Special Privileges and Licensing, earlier in this chapter.
- **Projects** lists all projects to which the user currently has access.
- **Email** specifies the user's E-mail address, enabling the user to send and receive E-mail from the Architect/Requirements client. For more information about this feature, see the *Systems Architect/Requirements Management User's Manual*.
- **Name** uniquely identifies the Architect/Requirements user name and applies to all transactions associated with the user. That association changes in the database when the user name is changed.
- **First Name** and **Last Name** describe additional user name information.

To modify a user profile:

1. In the **Users** folder for the project, select the user object.

   The properties table displays all viewable properties that apply to the user. Values that you cannot change are dimmed in the **Value** column.
2. Do any or all of the following:
   - To change the maximum privilege level for all projects:

.

Double-click the **Maximum Privilege** property value to display the Single-Choice dialog window.

. Check the checkbox for one of the following:

- **No Access** deactivates the user and revokes access to all projects.
- **Read Only** allows the user only to view objects in the Systems Engineering and Requirements Management module for all projects. The user cannot view schema objects in the Administration module.
- **Read and Write** allows the user to view, modify, create, and delete objects for all projects.
- **Power User** allows the user to perform certain project administration tasks in specific projects.
- **Project Administrator** allows any action in specific projects.
- **Enterprise Administrator** allows any action in all projects. You must have **Enterprise Administrator** privilege to see this value.

. Click **OK** to apply the changes and close the dialog window.

- To change the access privilege for this project:

.

Double-click the **User Access** property value to display the Single-Choice dialog window.

. Check the checkbox for one of the following:

- **No Access** revokes the user's access to this project.
- **Read Only** allows the user only to view objects in the Systems Engineering and Requirements Management module for this project. The user cannot view schema objects in the Administration module.
- **Read and Write** allows the user to view, modify, create, and delete objects in this project.
- **Power User** allows the user to perform certain project administration tasks in specific projects.
- **Project Administrator** allows any action in this project.
- **Enterprise Administrator** allows any action in all projects. You must have **Enterprise Administrator** privilege to see this value.

. Click **OK** to apply the changes and close the dialog window.

- To change special privileges for this project:

.

Double-click the **Additional Privilege** property value to display the Multi-Choice dialog window.

. Do one or both of the following:

- Check the checkbox for each privilege that you want to add, or click **Select All** to check all checkboxes simultaneously.
- Clear the checkbox for each privilege that you want to remove, or click **Unselect All** to clear all checkboxes simultaneously.

You can check or clear checkboxes in any combination, and you can use the checkboxes and buttons together.

. Click **OK** to apply the changes and close the dialog window.

- To change the projects to which the user currently has access:

.

Double-click the **Projects** property value to display the Multi-Choice dialog window.

. Do one or both of the following:

- Check the checkbox for each project to which you want to grant access, or click **Select All** to check all checkboxes simultaneously.
- Clear the checkbox for each project to which you want to revoke access, or click **Unselect All** to clear all checkboxes simultaneously.

You can check or clear checkboxes in any combination, and you can use the checkboxes and buttons together.

. Click **OK** to apply the changes and close the dialog window.

- To change the E-mail address:

. Double-click the **Email** property value to open a text field.

. Enter the E-mail address, and then press the enter key.

- To change the Architect/Requirements user name:

. Double-click the **Name** property value to open a text field.

. Enter the new user name, and then press the enter key.

In the content table, you can open a text field for the user object by pulling down the **File** menu and choosing **Rename**. You can also right-click the user object and choose **Rename** from the popup menu. Or, press the F2 key.

- To change additional user name information:

. Double-click the **First Name** or **Last Name** property value to open a text field.

. Enter the information, and then press the enter key.

Procedure Notes

Step 2: You can reverse any action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Resetting a User Password

Although users can change their own passwords in the Systems Engineering and Requirements Management module, you may need to reset a password in some situations. For example, a user who has forgotten the current password cannot supply that password before entering a new one, as is necessary in the Systems Engineering and Requirements Management module. Or, perhaps a user has left your team and you want to assign that user name to someone else, with a different password for security.

You must reset the password for a user who attempts to log in to Architect/Requirements with an expired password when Security Services is disabled. Any user whose password is reset must log in to Architect/Requirements with the new password to start the client or to continue the current session.

You enter the new password in the Change Password dialog window. The current password is not required.

The Administration module does not list user passwords.

To reset a user password:

1. In the navigation tree or the content table, open the **Users** folder for the project.

   The content table displays the users who have access to the project.

2. Select the user in the content table, pull down the **Tools** menu, and choose **Change Password**.

   Architect/Requirements displays the Change Password dialog window.

3. Enter the new password in the **New Password** and **Verify New Password** fields.
4. Click **OK**.

   This action clears the queue for the **Undo** option and cannot be reversed.

   The dialog window closes, and a confirmation message is displayed.

Creating a User Group

A *user group* consists of shortcuts to existing user objects in the project. By creating a user group and modifying it to add the users, you can define a collection of shortcuts to individual user objects. Then you can manage those users through the shortcuts in the user group, as if you selected the actual user objects. For more information, see Modifying a User Group, later in this chapter.

For a user represented by a shortcut, you can grant or revoke project access, modify the user profile, and reset the user password. For more information, see Granting or Revoking Project Access, Modifying a User Profile, and Resetting a User Password, earlier in this chapter.

To create a user group:

1. In the navigation tree or the content table, open the **Users** folder for the project.
2. Pull down the **File** menu and choose the **New→Group** options.

   This action cannot be reversed. A user group cannot be deleted from the database.

   The content table displays the new user group with a default name in an open text field.
3. Enter the user group name, and then press the enter key.

Procedure Notes

Step 2: You can also right-click the **Users** folder in the navigation tree, or right-click anywhere in the content table, and then choose the **New→Group** options from the popup menu. Or, click the **New User Group** button on the toolbar or press control-G.

Step 3: You can reverse this action by pulling down the **Edit** menu and choosing **Undo Rename**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Modifying a User Group

After creating a new user group, you modify the user group to add individual users. For an existing user group, you can add and remove users at any time.

Users are associated with user groups through shortcuts to the actual user objects. Consequently, a given user can belong to many user groups concurrently and can be managed from any one. Changes to a shortcut in a user group affect the actual user object and are reflected in all other user groups to which that user belongs. When you remove users, you delete only the shortcuts that associate those users with the user group, and the user objects themselves remain in the database.

Project Administrator privilege is not required if:

- Your user name is listed in the **Modify And Read Access** property value of the security profile that is applied to the user group.
- A maximum privilege level of **Read and Write** or higher is assigned to your user object in the **TcSE Administration** folder.

To add users to a user group:

1. In the **Users** folder for the project, select each user that you want to add.

   You can select nonadjacent and adjoining users.

   You can drop the selection on the intended user group to add these users and complete this procedure.

2. Pull down the **Edit** menu, and choose **Copy**.
3. Select the user group, pull down the **Edit** menu, and choose **Paste**.

   The content table displays the user object shortcuts below the user group.

Procedure Notes

Step 2: You can also right-click the selection and choose **Copy** from the popup menu. Or, click the **Copy** button on the toolbar or press control-C.

Step 3: You can also right-click the user group and choose **Paste** from the popup menu. Or, click the **Paste** button on the toolbar or press control-V. You can reverse this action by pulling down the **Edit** menu and choosing **Undo Copy**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

To remove users from a user group:

1. In the **Users** folder for the project, click the plus sign (+) for the user group, and then select each user that you want to remove.

   You can select nonadjacent and adjoining users.

2. Pull down the **File** menu and choose **Delete** to display a confirmation message, and then click **Yes** to delete the shortcuts from the database.

Procedure Notes

Step 2: You can also right-click the selection and choose **Delete** from the popup menu. Or, click the **Delete** button on the toolbar or press the delete key. To reverse this action, you can pull down the **Edit** menu and choose **Undo Delete**, click the **Undo** button on the toolbar, or press control-Z.

Deactivating a User

User objects cannot be deleted from the database. By deactivating a user, however, you can revoke the user's access to all objects in the database. The database retains the records of all previous transactions associated with that user name.

Deactivation also removes the association between the user name and the current license. That license can then be assigned to a new user, or the license can be held in reserve. For more information, see Licensing and Creating a New User, earlier in this chapter.

The user object is removed from the **Users** folder of each project to which the deactivated user had access. However, that user object remains in the **Users** folder of the **TcSE Administration** project, where project access can be granted to reactivate the user in the future if a license is available. For more information, see Granting or Revoking Project Access, earlier in this chapter.

To deactivate a user:

1. In the navigation tree or the content table, open the **Users** folder for the project.
2. In the content table, select the user.

   The properties table displays all viewable properties that apply to the user.
3.

   In the **Value** column of the properties table, double-click the value for the **Maximum Privilege** property.

   The Single-Choice dialog window is displayed.
4. Check the checkbox for **No Access**, and then click **OK** to close the dialog window.

   The deactivated user object is removed from all projects except the **TcSE Administration** project.

   For this project only, you can restore the user object and reactivate the user by immediately pulling down the **Edit** menu and choosing **Undo Change Properties**. You can also click the **Undo** button on the toolbar or press control-Z.

   Although the user object is retained in the **TcSE Administration** project, the user object is not restored in any other project, nor is the user reactivated for those projects.

Emptying a User's Recycle Bin

An Architect/Requirements enterprise administrator can empty a selected user's Recycle Bin for system management purposes. For example, a Recycle Bin that is consuming excess space can be emptied to increase storage capacity in the database. Also, the administrator can empty a Recycle Bin to remove deleted instances of an object type and free the corresponding type definition for deletion.

A Recycle Bin can be emptied whether or not the user is logged on to Architect/Requirements.

Objects cannot be restored after a Recycle Bin is emptied. The objects are permanently removed from the database.

- You must have **Enterprise Administrator** access privilege to empty a user's Recycle Bin.
- An enterprise administrator can set the number of days after which all users' Recycle Bins are emptied automatically. This setting is the value of the **DB.RecycleBinTimeout** parameter, under **Web Application Configuration** in **Administrative Tools**. For more information about customizing the Architect/Requirements server, see the *Systems Architect/Requirements Management System Administrator's Manual*.

To empty a user's Recycle Bin:

1. In the **TcSE Administration** project, open the **Users** folder.
2. Select the user in the content table, pull down the **Tools** menu, and choose **Empty Recycle Bin**.

   Architect/Requirements displays a message asking you to confirm this action.
3. Click **Yes**.

   This action clears the queue for the **Undo** option and cannot be reversed.

   Architect/Requirements removes the objects from the database.

Chapter 6: Maintaining Project Security

This chapter contains an overview of security profiles and instructions for using these profiles to define rules for user permissions within a project.

Introduction

Chapter *5, Managing Users,* described access rights that can be granted to users at the project level. This chapter describes access rights that can be defined at the object level using security profiles. For a user, the project-level access rights are enforced first and can be limited by security profiles, but cannot be extended. For example, if a user has been granted **Read Only** access to a project, then no security profile can grant that user modification rights to any object within the project.

Security Profiles and Access Control

A security profile is a set of access rules that control user actions on specific objects in Architect/Requirements. Security profiles can be applied to objects in the Systems Engineering module and the Administration module, controlling which users can view, modify or delete the objects. A given security profile can be applied to any number of objects.

A security profile defines the users who can work with the objects to which the security profile applies. Each user's actions on those objects are determined by the permission granted to that user. User permissions can be granted individually, by specific user names, and generally, through keywords relative to all user names.

Security profiles can be automatically applied when an object is created. Applying security profile when an object is created prevents users from creating objects of a certain type or with a certain owner. If a new object inherits a security profile from its owner or gets the **Instance Security Profile** from its type definition, that security profile takes effect when the object is created. If the created users do not have at least write access in the profile, they are not allowed to create the object. If **Creator** is specified in the profile's **Full Control** or **Modify And Read Access** list, any user is able to create the object.

Individual users can be granted the following permissions:

- **Full Control** allows users to view, modify, and delete the objects. These users can also change the security profile itself.
- **Modify And Read Access** allows users to view and modify the objects. These users cannot delete the objects or change the security profile itself.
- **Read Access** allows users only to view the objects. However, these users can attach notes and create trace links to the objects if they have **Read and Write** access to the project as a whole.

Enterprise administrators have full access to all the objects in all projects, irrespective of security profiles. Users with **Project Administrator** rights in a project have full access to all the objects in that project, irrespective of security profiles.

By default, a user who is not granted any of those permissions cannot work with the objects. For that user name, the objects do not appear in the Systems Engineering module.

In addition, each permission can be granted generally by the following keywords:

- **Creator** grants the permission to the user who originally created the object.
- **Everyone** grants the permission to all users in the project.
- **No One** revokes the permission for all users in the project.

  Siemens PLM Software recommends against using the **No One** keyword for the **Read Access** permission.

Security profiles can be created as needed. However, it is best to manage security profiles in a structured way that is consistent with your configuration management processes. For example:

- By project phase.

  Create a security profile for each phase of the project life cycle, defining each profile with access control appropriate for that phase. For example, security profile names may correspond to project phases such as Proposal, Preliminary Design, Post-PDR, Post-CDR, and Released, with access control becoming more stringent for each successive phase.

  As each object moves through the life cycle, the object's security profile setting is changed to the profile for the current phase. The security profiles themselves do not change. In addition, different types of objects may have different life cycles, and each type may have its own cycle of security profiles.

  Users can change the security profile setting for an object in the Systems Engineering and Requirements Management module by editing the object's **Security Profile** property. For more information about editing properties, see the *Systems Architect/Requirements Management User's Manual*.

- By object relationships.

  Create a common security profile for a set of closely related objects that move together through the project life cycle. For example, a separate security profile with unique access control may be defined for each of three requirement subtypes, such as Customer Requirements, Functional Requirements, and Design Requirements.

  The security profile for a subtype is applied to those objects as they are created. As the entire set passes from one phase, the shared security profile is modified for the next phase by the project administrator.

  With this method, it is not necessary to change the security profile settings for individual objects. Also, static inheritance can be used to apply security profiles to new objects automatically. For more information, see Inherited Security and Setting Access Control Inheritance, later in this chapter.

Inherited Security

Where new objects are managed under the same access control as their superiors, common use cases call for inheritance. A property of security profiles, **Apply To New Descendents**, can specify that new objects reference this property value in the owner's security profile.

If the value is **MEMBER**, the new object is set to point to that same security profile, and the setting is shown in the **Security Profile** property value for the new object. That behavior applies to newly created objects, including objects that are created by copying existing objects.

When an object is created through a copy operation, the new object does not take its security profile setting from the originating object. Instead, the setting for the copy depends on the **Apply To New Descendents** property value of the security profile for the copy's owner:

- If the value is **MEMBER**, the copy inherits that security profile.
- If the value is **None**, the copy receives a blank value in its **Security Profile** property.

The behavior for move operations is more complicated, because descendants of a moved object are also moved to preserve the hierarchy:

- A moved object inherits the security profile of its new owner if:
    - The moved object and its previous owner point to the same security profile.
    - For that security profile, the **Apply To New Descendents** property value is **MEMBER**.
    - The new owner points to a security profile whose **Apply To New Descendents** property value is **MEMBER**.

  Otherwise, the moved object keeps its current security profile, if any.
- For descendants of a moved parent object:
    - If a descendant points to the same security profile as its parent before the move, the descendant inherits the security profile of the parent's new owner.
    - If a descendant and the parent point to different security profiles before the move, the descendant keeps its current security profile, if any. If that descendant is itself a parent, each lower level object keeps its current security profile, if any.

To track inheritance, Architect/Requirements creates a text property named **Security Profile Reason** for the object. This property value is a text string that explains how the security profile became associated with the object.

For a moved object, for example, the value may provide information such as:

```
Inherited from superior at Move <create user name> hh:mm mm/dd/yy
```

Also, a security profile applied directly to the object may produce a value such as:

```
Set by user <change user name> hh:mm mm/dd/yy
```

The value is overwritten each time the object's security profile is changed.

Security Profile Support for Schema Objects

As with all objects in the Systems Engineering and Requirements Management module, Architect/Requirements supports the security profile mechanism on schema objects in the Administration module. Except for user objects, security profiles can be applied to all schema objects. The project administrator can apply a security profile to specific schema objects to allow certain users to modify those objects in the Administration module. For more information, see Applying a Security Profile to a Schema Object, later in this chapter.

Security profiles do not keep users from viewing objects in the Administration module. To prohibit schema object visibility for a certain user, assign the **Read Only** access privilege to the **User Access** property of the user object. For more information, see Project Access and Modifying a User Profile in chapter 5, *Managing Users*.

To create activators in a project, users must have **Project Administrator** privilege and the additional **Script Authoring** privilege for the project. Users can modify an existing activator if they have **Script Authoring** privilege for the project and **Modify** permission for the activator.

Access Control for Object Properties

For a property definition in the Administration module, the **Instance Security Profile** property monitors access to all instances of that property in the Systems Engineering and Requirements Management module, regardless of the object types on which the instances appear.

- The value of the **Instance Security Profile** property is the name of a specific security profile. This security profile governs user access to all instances of the property.

  When the value is blank, access to instances of this property is checked using the owner's security profile. The value is blank by default.

- User access to property instances is checked dynamically using the **Instance Security Profile** value on the property definition. The value is not copied onto every instance of the property.

- If the **Instance Security Profile** value is set for a property definition, that security profile overrides each owning object's **Security Profile** settings for instances of that property. Access rights of the two security profiles are not combined in any way. When a property's **Instance Security Profile** is set, there is no way for any object to apply any different level of access to its own value for that property.

- The **Instance Security Profile** value determines only those users who can modify the property instances. The value does not control access to view the property instances. If a user has **Read Access** to an object, the user can view all properties for an object.

Creating a Security Profile

A security profile defines the users who can work with the objects to which the security profile applies. Each user's actions on those objects are determined by the permission granted to that user.

After you create a security profile, you follow up by editing the security profile to specify users and permissions. For more information, see Setting User Permissions, later in this chapter.

To create a security profile:

1. In the navigation tree or the content table, open the **Security Profile** folder for the project.
2. Pull down the **File** menu and choose the **New→Security Profile** options.

   The content table displays the new security profile, with the temporary name in an open text field.
3. Enter a meaningful name for the security profile, and then press the enter key.

   The content table displays the security profile at the bottom of the pane. To place the security profile according to the sort sequence, pull down the **View** menu and choose **Refresh**. Or, right-click the root node, and then choose **Refresh** from the popup menu.

Procedure Notes

Step 2: You can also right-click the **Security Profile** folder and choose the **New→Security Profile** options from the popup menu.

Setting User Permissions

A security profile defines user permissions for the objects to which the security profile applies. Permissions correspond to the following properties of the security profile:

- **Full Control** identifies the users who can view, modify, and delete the objects. These users can also change the security profile itself.
- **Modify And Read Access** identifies the users who can view and modify the objects. These users cannot delete the objects or change the security profile itself.
- **Read Access** identifies the users who can only view the objects. However, these users can attach notes and create trace links to the objects.

To set user permissions:

1. In the **Security Profiles** folder or subfolder, select the security profile.
2. In the **Value** column of the **Properties** tab or window, double-click the value for a permission to display the Multi-Choice dialog window.
3. Do one or both of the following:
   - To set this permission for individual users:
     - Check the checkbox for a user to grant the permission.
     - Clear the checkbox for a user to revoke the permission.
   - To set this permission generally by a keyword:
     - Check the checkbox for **Creator** to grant the permission to the user who created the object. Or, clear the checkbox to remove this keyword.
     - Check the checkbox for **Everyone** to grant the permission to all users in the project. Or, clear the checkbox to remove this keyword.
     - Check the checkbox for **No One** to revoke this permission for all users in the project. Or, clear the checkbox to remove this keyword.

     
     Siemens PLM Software recommends that you do not set **No One** as the only value of the **Read Access** property.

   
   - If you check **Everyone** and one or more user names or other keywords, the permission is granted to all users in the project. The user names and other keywords are ignored.
   - If you check **No One** and one or more user names or **Creator**, and if **Everyone** is not checked, the permission is granted to the specified users. **No One** is ignored.
4. Click **OK**.

   The property value reflects the net result of your selections.

   To set an additional permission, repeat steps 2 and 3. To reverse the setting, you can pull down the **Edit** menu and choose **Undo Change Properties**, click the **Undo** button on the toolbar, or press control-Z.

Setting Access Control Inheritance

For new members of an existing object, access control can be set automatically through the **Apply To New Descendents** property of the owner's security profile. When the property value is **MEMBER**, all new members of the object inherit the same security profile.

Later changes to the owner's security profile setting do not automatically propagate down to the object's descendants. To change access control throughout an existing object hierarchy, the security profile setting of each object must be changed individually. To see what controls are in effect for an object, users can view the properties of the applicable security profile in the **Security Profiles** folder for the project. For more information about editing properties, see the *Systems Architect/Requirements Management User's Manual*.

Inheritance for moved objects involves additional considerations. For more information, see Inherited Security, earlier in this chapter.

- A security profile must exist in the project. For more information, see Creating a Security Profile, earlier in this chapter.
- In some cases, a user who creates an object may not be allowed to edit the new object's properties. To avoid this situation, include the **Creator** keyword in the **Modify And Read Access** property of the security profile that you assign to the owning object. For more information, see Setting User Permissions, earlier in this chapter.
- Access control inheritance overrides a default security profile. For more information, see Assigning a Default Security Profile to New Objects of a Type, later in this chapter.

To set access control inheritance:

1. In the **Security Profiles** folder, select the security profile that you want to assign to the owning object, or open the containing subfolder and then select the security profile.

   The **Properties** tab or window displays all viewable properties for the security profile.

2.

   In the **Value** column, double-click the **Apply To New Descendents** property value to display the Single-Choice dialog window.

3. Check the **MEMBER** checkbox, and then click **OK** to close the dialog window and set the property value.

4. In the Systems Engineering module, select the owning object, and then set the object's **Security Profile** property value to the security profile selected in step 1.

Assigning a Default Security Profile to New Objects of a Type

Through the **Instance Security Profile** property of a type definition, you can limit user access for all new objects of that type. In the property value, the default security profile that you assign is automatically applied to new objects that are subsequently created in the Systems Engineering module, including those created by copy and import operations.

The **Instance Security Profile** property is a system-defined property, whose initial value is blank. When a new object is created, the property value of the corresponding type definition is referenced. If the property value names a security profile, that security profile is applied to the new object by default. The default security profile can be changed by users in the Systems Engineering module.

When a subtype is created, it inherits the security profile named in the **Instance Security Profile** property for the parent type definition. Therefore, all new objects of the new subtype receive the security profile in the Systems Engineering module. For more information, see Creating a Subtype in chapter 3, *Customizing Object Types*.

- A security profile must exist in the project. For more information, see Creating a Security Profile, earlier in this chapter.
- In some cases, a user who creates an object may not be allowed to edit the new object's properties. To avoid this situation, include the **Creator** keyword in the **Modify And Read Access** property of the security profile that you assign to the type definition. For more information, see Setting User Permissions, earlier in this chapter.
- Access control inheritance overrides the default security profile specified in the **Instance Security Profile** property. For more information, see Setting Access Control Inheritance, earlier in this chapter.

To assign a default security profile to new objects of a type:

1. In the navigation tree or the content table, open the project's **Type Definitions** folder.

   The content table displays the base type definitions. To see lower level subtypes in a hierarchy, click the plus signs in the **Types/SubTypes** column.

2. In the content table, select the type definition.

   The **Properties** tab or window displays all viewable properties for the type definition.

3.

   In the **Value** column, double-click the **Instance Security Profile** property value to display the Single-Choice dialog window.

4. Check the checkbox for the security profile, and then click **OK** to close the dialog window and set the property value.

   To reverse this action, you can pull down the **Edit** menu and choose **Undo Change Properties**, click the **Undo** button on the toolbar, or press control-Z.

Applying a Security Profile to a Schema Object

If a security profile is not applied to a given schema object, only users with **Project Administrator** privilege can modify that object. By applying a security profile, you can give certain users permission to modify the object, without designating additional project administrators.

Security profiles do not keep users from viewing objects in the Administration module. To prohibit schema object visibility for a certain user, assign the **Read Only** access privilege to the **User Access** property of the user object. For more information, see Project Access and Modifying a User Profile in chapter 5, *Managing Users*.

You can apply a security profile to any schema object in the following primary folders:

- **Activators**
- **Property Definitions**
- **Reports and Formatting**
- **Security Profiles**
- **Type Definitions**

In the **Users** folder, security profiles can be applied to user groups to enforce access control on those objects. However, a security profile has no effect when applied to an individual user object.

To apply a security profile to a schema object:

1. In the navigation tree, open the primary folder that contains the object.

   The content table displays the schema objects and subfolders in the primary folder.

2. In the content table, select the object to which you want to apply the security profile.

   The **Properties** tab or window displays all viewable properties for the object.

3.

   In the **Value** column, double-click the **Security Profile** property value to display the Single-Choice dialog window.

4. Check the checkbox for the security profile, and then click **OK** or press the enter key to close the dialog window and set the property value.

   You can reverse this action by pulling down the **Edit** menu and choosing **Undo Change Properties**, by clicking the **Undo** button on the toolbar, or by pressing control-Z.

Chapter 7: Configuring the Change Management Package

This chapter provides an overview of the Architect/Requirements change management package and contains instructions for setting up the change approval process and for activating change logs for type definitions.

Change Approval

From the Systems Engineering module, requirements and building blocks can be submitted through the Architect/Requirements change management package for change approval. A submitted object may be either of the following:

- An object with no prior versions is a new submission. When the changes are submitted, the object is frozen for the first time.
- An object with prior versions is a revision to previously submitted content. When the changes are submitted, this revision is frozen.

To use the change approval process, the **Version** package must be enabled for the project. For more information about enabling versions for a project, see Enabling the Versions Package for the Project, later in this chapter.

The change approval process tracks the following events:

- A **Submit** event occurs when a requirement or a building block is submitted for approval. E-mail is sent to recipients in the following categories:
  - *Approvers* have the responsibility to approve or reject the changes. Each of these users receives an E-mail message containing two hyperlinks:
    - A hyperlink to the object in the Architect/Requirements client. The user can click this hyperlink to navigate to the object and review the changes.
    - A hyperlink to the change management package. The user can click this hyperlink to navigate to the approval and rejection page in Microsoft Internet Explorer.
  -

Notifiers do not approve or reject the changes, but are notified of submitted changes for informational purposes. Each of these users receives an E-mail message containing a hyperlink to the submitted object in the Architect/Requirements client. The user can click the hyperlink to navigate to the object and review the changes.

Each message also contains the submitter's comments, if any. In addition, each approver and notifier receives the object's content in **.mhtml** format as an E-mail attachment. For an object that has prior versions, the previously submitted content is included.

The **Submit** event automatically creates a *change approval object* for the submitted object. Displayed in the **Attachments** tab or window, the change approval object contains a table of information related to the approval process. The initial table row is added for the **Submit** event.

- A **Response** event occurs each time an approval or a rejection is received.

  For each response, a row is added to the change approval object that is attached to the submitted object. Delinquent responses are also tracked.

- An **Approved** event occurs if the changes are approved by all approvers. The approved object remains frozen. An E-mail message is sent to each approver and to the submitter, stating that the changes are approved.

  The **Approved** event concludes the approval process, and a final row is added to the change approval object.

- A **Rejected** event occurs each time the changes are rejected by one approver. When the first **Rejected** event occurs, the rejected object is unfrozen and remains in that state.

  For each **Rejected** event:

  - An E-mail message is sent to the submitter only, stating that the changes are rejected by the individual approver.
  - A row is added to the change approval object for this event.

  If other **Response** events are received after the first rejection, each later event is added to the change approval object.

When changes are submitted, an E-mail message is sent to the approvers and notifiers. The approvers and notifiers must be identified in a security profile, and that security profile must be applied to the submitted object in the Systems Engineering module. Users can apply the security profile through the object's **Security Profile** property in the **Properties** tab or window.

Each approver automatically has full access to the changes until that user approves or rejects the change request.

A change approval object is automatically generated for a requirement or a building block that is submitted for approval. The change approval object contains a table in which the rows show all responses that are currently received. Each row shows the time of response, the responding user name, the user's comments, and the response type. A new row is added for each subsequent approval or rejection. If the changes are approved, a final row is added for that event. The change approval object is displayed in the **Attachments** tab or window in the Systems Engineering module.

The change management package can be customized with the Architect/Requirements application programming interface (API). For more information, see the *Systems Architect/Requirements Management API Reference*.

Enabling the Versions Package for the Project

In the Administration module, the **Packages** property for the project determines whether versions are enabled. When the **Version** value is added, users can do the following in the Systems Engineering and Requirements Management module:

- Create versions of requirements, building blocks, and their subtypes, and also create versions of prior versions of those objects.
- Choose effectivity rules through controls displayed in the **Address** bar.
- View a version tree in the **Versions** tab and window in the notebook pane.
- Create baselines of requirements, building blocks, and their subtypes.
- View a baseline in the hierarchical content table.

For more information about working with versions, see the *Systems Architect/Requirements Management User's Manual*.

Once the **Version** package is added, it cannot be removed from the project.

To enable the Versions package for the project:

1. In the navigation tree, select the **Projects** node.

   The content table displays all projects to which you have access.

2. Select the project in the content table.

   The **Properties** tab or window displays all viewable properties for the project.

3.

   In the **Value** column, double-click the **Packages** property value.

   The Multi-Choice dialog window is displayed.

4. Check the **Version** checkbox, and then click **OK** to close the dialog window.

   A confirmation message states that the **Version** package cannot be removed, and asks if you want to continue.

5. Click **Yes**.

   This action cannot be reversed. The **Version** package cannot be removed from the project.

   The **Version** value is added to the **Packages** property, indicating that versions are enabled for the project.

Enabling Effectivity-Sensitive References for a Project

A reference to a versionable object can be either fixed or dynamic. Fixed references always get the information from the same version of the object. Dynamic references get the information from the currently effective version of the object. For more information about the behavior of reference links, see *Systems Architect/Requirements Management User's Manual*.

There are two ways to specify how dynamic links behave. In the Administration module, the **Project Settings** property of a project controls the default behavior for reference links in that project. Selecting the **Promote References** option indicates dynamic references. By default, a project uses fixed references.

The behavior for a specific link can be controlled with the **Reference Settings** property (which can be accessed by right-clicking a reference link). Double-clicking the **Reference Settings** property allows you to select or deselect the **Promote References** value. The **Reference Settings** property allows a specific reference link to override the default behavior.

The use of fixed or dynamic references does not affect how the references are stored in the database. The setting for a project can be changed at any time to change the behavior of the existing references.

Although the **Reference Settings** property is a multi-select property, the **Promote References** and **Fixed References** options are mutually exclusive. If you select both the options, **Promote References** takes precedence.

The **Reference Settings** property requires **Read & Write** privileges to the objects for which the reference link is created. When the **Project Settings** property is set to **Promote References**, the **Reference Settings** property on a reference link is displayed as blank. In this case, enabling or disabling **Promote References** does not matter because the project-wide behavior of **Promote References** is honored.

The **Promote References** behavior is supported for references to versionable objects, requirements, buildings blocks, and shortcuts to versionable objects. References to notes do not support the **Promote References** behavior.

When the **Promote References** option is enabled, reference links are sensitive to effectivity. The symbolic references track the effective version of the referenced object. This behavior occurs whether the referencing object is frozen or not. When effectivity is changed, the symbolically referenced information changes to be consistent with the new effectivity. The effectively referenced version is used when a requirement is opened in Microsoft Office Word, shown in the preview pane, or exported to a document.

For example, consider a frozen object A referencing a non-frozen object B. B1 is created as the new version of B. The similarities and differences in the resolution of reference links with and without enabling the **Promote References** option setting can be seen in the **Links→Relations** and the **Where Used** tabs in the notebook pane.

Table 7-1 describes the resolution of the reference links with the **Promote References** option disabled. Table 7-2 describes the resolution of the reference links with the **Promote References** option enabled.

Table 7-1. Reference Links with Promote References Disabled

Tab in the Notebook pane	Effectivity: Current Version	Effectivity: Current Frozen Version (same behavior)
Links→Relations	Selecting A displays B as the target.	Selecting A displays B as the target.

Table 7-1. Reference Links with Promote References Disabled

	Selecting B1 displays no reference.	Selecting B displays A as the source.
Where Used	Selecting B1 displays no reference.	Selecting B displays A.

Table 7-2. Reference Links with Promote References Enabled

Tab in the Notebook pane	Effectivity: Current Version	Effectivity: Current Frozen Version (same behavior)
Links→Relations	Selecting A displays B1 as the target. Selecting B1 displays A as the source.	Selecting A displays B as the target. Selecting B displays A as the source.
Where Used	Selecting B1 displays no reference.	Selecting B displays A.

Setting Up the Change Approval Process

The change approval process uses E-mail for notification of change submissions, approvals, and rejections. Therefore, the following E-mail information must be supplied:

- The IP address of your E-mail server.

  The E-mail server IP address is a Web application configuration parameter of the Architect/Requirements administrative tools. You gain access to these tools through Microsoft Internet Explorer.

  Enterprise Administrator privilege is required to enter the E-mail server IP address and all other configuration parameters.

- A valid E-mail address for each approver, notifier, and submitter. You enter these E-mail addresses in the **Users** folder for the project in the Administration module.

In a security profile, you designate users and user groups in the following categories:

- Approvers have the responsibility to approve or reject changes.
- Notifiers only receive notification of submitted changes for informational purposes.

You can create a new security profile especially for this change approval process. Or, you can modify an existing security profile. For more information, see Creating a Security Profile or Setting User Permissions in chapter **Unsatisfied xref reference**, ***Unsatisfied xref title***.

Each approver automatically has full access to the changes until that user approves or rejects the change request.

The security profile must be applied to each requirement and building block that is submitted for change approval in the Systems Engineering and Requirements Management module. Users can apply the security profile through the object's **Security Profile** property in the **Properties** tab or window.

You must enable the **Version** package for the project before changes can be submitted for approval. For more information, see Enabling the Versions Package for the Project, earlier in this chapter.

To set up the change approval process:

1. To enter the E-mail server IP address, do the following in Internet Explorer:

   You must have **Enterprise Administrator** privilege to perform this step.

   a. Open the Architect/Requirements home page, and then click the **Administrative Tools** hyperlink.

      The Administrative Tools page is displayed.

   b. On the Administrative Tools page, click the **Web Application Configuration** hyperlink.

      The Architect/Requirements log in page is displayed.

   c. Enter your Architect/Requirements user name and password, select a language, and click **Log In**.

      The Architect/Requirements Configuration Parameters page is displayed.

   d. In the **MailServerIP** field, enter the E-mail server IP address.

   e. At the bottom of the page, click the **Update** button.

      The Architect/Requirements Configuration Parameters page is displayed with read-only information for verification.

   f. Do one of the following:

      - To verify that the E-mail server IP address is correct, click the **Ok** button.

        The Architect/Requirements Configuration Parameters page is displayed, and the E-mail server IP address is entered for the change approval process.

      - To change the E-mail server IP address, click the **Back** button to redisplay the fields on the Architect/Requirements Configuration Parameters page. Then repeat steps d through f.

2. To enter E-mail addresses for approvers, notifiers, and submitters, do the following in the Administration module:

   a. In the navigation tree, select the **Users** folder for the project.

      The content table displays all users in the project.

   b. Do the following for each approver, notifier, and submitter:

- Select the user object in the content table.

  The **Properties** tab or window displays all viewable properties for the user.
- In the **Value** column, double-click the **Email** property value to open a text field.
- Enter the user's E-mail address in the text field, and then press the enter key.

Also in the **Users** folder, you can give users permission to modify the activators related to change approval process. For each of these users, select the user object in the content table, and double-click the **Additional Privilege** property value to display the Multi-Choice dialog window. Check the checkbox for **Script Authoring**, and then click **OK** or press the enter key.

In the **Activators** folder, the related activators are the following:

c.

Change Submit executes when a user submits changes for approval in the Systems Engineering and Requirements Management module.

d.

Change Response executes when an approval or a rejection is received.

e.

Change Approved executes when all approvers listed in the security profile have approved the changes.

f.

Change Rejection executes when any one approver rejects the changes.

3. To designate the approvers and notifiers, do the following in the Administration module:
   a. In the navigation tree, select the **Security Profiles** folder for the project.

      The content table displays all security profiles in the project.
   b. In the content table, select the security profile that you want to use for this change approval process.

      The **Properties** tab or window displays all viewable properties for the security profile.
   c. Do one or both of the following:
      - To designate approvers:
        c.

        In the **Value** column, double-click the **Change Approvers** property value to display the Multi-Choice dialog window.

        c. Check the checkbox for each user and user group that you designate as an approver, and clear the checkbox for each user and user group that you want to exclude.

           You can use the checkboxes in conjunction with the buttons. Click **Select All** to check all checkboxes simultaneously. Click **Unselect All** to clear all checkboxes simultaneously.

        c. To close the dialog window and apply your choices, click **OK** or press the enter key.

Each approver automatically has full access to the changes until that user approves or rejects the change request.

- To designate notifiers:

  c.

  In the **Value** column, double-click the **Change Notifiers** property value to display the Multi-Choice dialog window.

  c. Check the checkbox for each user and user group that you designate as a notifier, and clear the checkbox for each user and user group that you want to exclude.

  You can use the checkboxes in conjunction with the buttons. Click **Select All** to check all checkboxes simultaneously. Click **Unselect All** to clear all checkboxes simultaneously.

  c. To close the dialog window and apply your choices, click **OK** or press the enter key.

Notifiers must have a minimum privilege of **Read Access** for the security profile. If necessary, double-click the **Read Access** property value and check the checkbox for each designated notifier who does not have the minimum privilege.

Change Logs

A change log captures the history of modifications to an object of a given type definition. Change logs can be activated for folders, requirements, building blocks, groups, notes, trace links, connections, diagrams, and spreadsheets.

For each type definition and user-defined subtype, the project administrator can specify the change events that are tracked by the change log. In the Systems Engineering module, the change log is automatically created for an object of that type when a specified change event occurs. For the object selected in the content table, the change log is displayed in the **Attachments** tab and floating window.

The change log contains a table in which the rows show all change events that are currently recorded. Each row shows the date and time of change, the user who made the change, and the type of change. For object property changes that are written to the change log, the log also shows the previous and new values. A row is added for each later change event for that object.

- The date and time shown for each change reflects the time zone and locale of the Architect/Requirements server. This time zone and locale may differ from that of the Architect/Requirements client.
- Change logs can be edited only by Architect/Requirements enterprise administrators and project administrators.

Enabling Change Logs for the Project

Change logs must be enabled for the project before change logs can be generated in the Systems Engineering module. You enable change logs in the Administration module by adding the **Change Log** value to the **Packages** property for the project. After you set up change logs for type definitions, modifications to the related objects can be tracked in the Systems Engineering module.

To enable change logs for the project:

1. In the navigation tree, select the **Projects** node.

   The content table displays all projects to which you have access.

2. Select the project in the content table.

   The **Properties** tab or window displays all viewable properties for the project.

3.

   In the **Value** column, double-click the **Packages** property value.

   The Multi-Choice dialog window is displayed.

4. Check the **Change Log** checkbox, and then click **OK**.

   The dialog window closes, and the **Change Log** value is added to the **Packages** property, indicating that change logs are enabled for the project.

Use change logs cautiously because they add database and processing overhead. To minimize the overhead, the administration controls for enabling change logging are very fine-grained. Enable change logging only on the object types where detailed change tracking has a necessary business purpose. For each of those types, enable change logging only for the events and properties that are essential for that type.

Setting Up Change Logs for a Type Definition

Change logs allow you to specify which change events are tracked for all objects of a given type definition. For an object of that type in the Systems Engineering and Requirements Management module, a change log is generated automatically when the first specified change event occurs. A row is added to the change log for each specified change event that subsequently occurs. Users can view the change log in the **Attachments** tab or floating window. For more information about viewing a change log, see the *Systems Architect/Requirements Management User's Manual*.

Change logs must be enabled for the project before change logs can be generated in the Systems Engineering module. After you enable change logs, you can set up change logs for folders, requirements, building blocks, groups, notes, trace links, connections, diagrams, and spreadsheets. Also, you can set up change logs for system-defined and user-defined subtypes. For more information, see Overview of Type Definitions and Creating a Subtype in chapter 3, *Customizing Object Types*.

To set up change logs for a type definition:

1. In the navigation tree or the content table, open **Type Definitions** folder for the project.

   The content table displays the base type definitions. To see lower level subtypes in a hierarchy, click the plus signs (+) in the **Types/SubTypes** column.

2. In the content table, select the type definition.

The **Properties** tab or window displays all viewable properties that apply to the type definition.

3. In the **Value** column, double-click the **Change Log** property value.

   The Multi-Choice dialog window displays the change events that are defined for the project.

4. Check the checkbox for each change event that you want to add to the change log.

   To remove a single change event, clear the checkbox.

   You can use the checkboxes in conjunction with the buttons:

   - Click **Select All** to check all checkboxes simultaneously.
   - Click **Unselect All** to clear all checkboxes simultaneously.

5. To close the dialog window and apply your choices, click **OK**.

   The change events are added to the **Change Log** property value for the type definition.

Change Event Flags

Below is a list of the system defined change events.

Event	Description
Add Complying Event	The end object for a new trace link or connection
Add Defining Event	The start object for a new trace link or connection
Add Incoming Event	A generic link has been created ending at this object
Add Member Event	A child is added to a container
Add Note Event	A note or other attachment is added
Add Outgoing Event	A generic link has been created starting from this object
Add Where Used Event	A short cut or group membership link is created to an object
Copy Object Event	An object is created by a copy operation
Delete Object Event	An object is deleted
Freeze Event	A requirement or building block is frozen
Modify Name Event	An object's name is changed
Modify ROIN Event	A requirement's ROIN is changed
Modify Security Event	A Security Profile property is set
Modify Text Event	The text content of a requirement or note is changed
Modify Type Definition Event	The Subtype of an object is changed
Move Object Event	An object is moved to a new owner
New Object Event	An object is created
New Variant Event	A variant of a requirement or building block is created

New Version Event	A version of a requirement or building block is created
Remove Complying Event	A trace link or connection to this object is deleted or removed
Remove Defining Event	A trace link or connection from this object is deleted
Remove Incoming Event	A generic link starting from this object has been deleted
Remove Member Event	An object is deleted or moved from a container
Remove Note Event	A note or other attachment is deleted or moved
Remove Outgoing Event	A generic link ending at this object has been deleted
Remove Where Used Event	A short cut or group membership link is deleted from an object
Restore Object Event	Object is restored from the recycle bin
Unfreeze Event	A requirement or building block is frozen

In addition to the system defined events all property instances defined for the selected type definition will appear as change events. These events are triggered whenever an instance of that property is modified. The list of property instances can be seen in the type definition's *Properties* property.

Processing a large number of change event can be time consuming and may hurt performance. Only set change flags which are needed.

Transaction Management

Requirements Service Transaction Logging

Architect/Requirements has an enhancement to the log server functionality that adds trace records for transaction start and end events associated with requirements service method invocations. You can use this information for tracking down issues related to server usage or performance. The following example shows the trace records associated with an invocation of the requirements service's **setObject** method:

Time(mm:dd:yyyy; hh:mm:ss:ms)	Thread	UserName	SessionName	className	originatingMethod	Level	Message
11-06-2012 23:42:27:146	http-bio-8080-exec-10	eriadm	session0	setObject	Transaction Started	TRACE	USER_PREFERENCES
11-06-2012 23:42:27:148	http-bio-8080-exec-10	eriadm	session0	setObject	Transaction End	TRACE	Success, USER_PREFERENCES

In the example, the following information is displayed:

Time(mm:dd:yyyy; hh:mm:ss:ms) Time on the Architect/Requirements server when the event was logged.

Thread Thread in the web server that is hosting the Architect/Requirements application. For example, Tomcat, WebSphere, or any other application server according to your configuration.

UserName	Architect/Requirements user responsible for the current transaction.
SessionName	Versant session name. It allows you to connect message with Versant administrator commands such as **db** tools. This information helps you in troubleshooting Versant issues.
ClassName	Requirement service method name.
originatingMethod	API called in the transaction.
Level	Distinguishes the message from other messages being logged. It is **TRACE** for such messages.
Message	Contains selected parameter information about the call. They provide additional information regarding the transaction. The end transaction method indicates success or failure for the transaction.

To enable the requirements service transaction logging, set the web application configuration parameter **Log.Server.Trace** to **true**.

It is not necessary to restart the Architect/Requirements application server to enable or disable this feature.

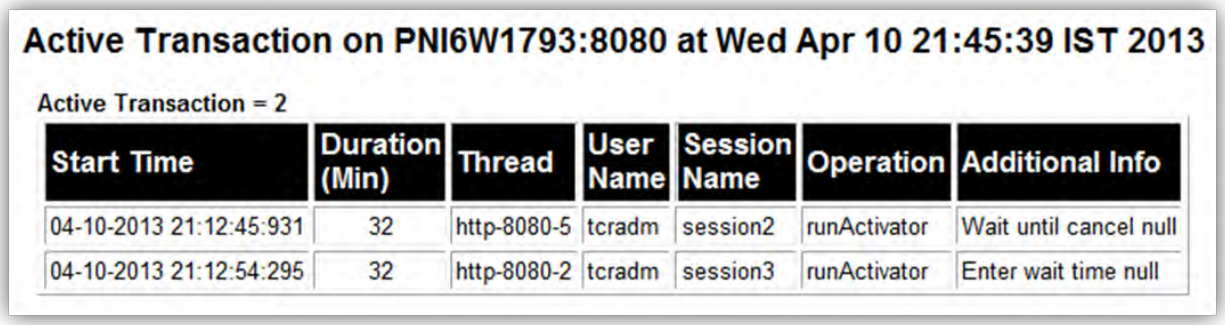
Requirements Service Transaction Monitoring

Architect/Requirements allows you to monitor the current requirements service transactions.

To view a snapshot summary of requirements service transactions currently in progress:

1. On the Architect/Requirements home page, click **Administrative Tools**.
2. On the **Administrative Tools** page, click **Diagnostic Tools**.
3. On the **Diagnostic Tools** page, scroll down and click **List Current Transactions**.

Following is an example of the **List Current Transactions** page:



Active Transaction on PNI6W1793:8080 at Wed Apr 10 21:45:39 IST 2013

Active Transaction = 2

Start Time	Duration (Min)	Thread	User Name	Session Name	Operation	Additional Info
04-10-2013 21:12:45:931	32	http-8080-5	tcradm	session2	runActivator	Wait until cancel null
04-10-2013 21:12:54:295	32	http-8080-2	tcradm	session3	runActivator	Enter wait time null

The following information is displayed about the current transactions:

Start Time	Time on the Architect/Requirements server when the transaction started.

Duration (Min)	Duration (in minutes) for which the transaction is running.
Thread	Thread in the web server that is hosting the Architect/Requirements application. For example, Tomcat, WebSphere, or any other application server according to your configuration.
User Name	Architect/Requirements user responsible for the current transaction.
Session Name	Versant session name. It allows you to connect message with Versant administrator commands such as **db** tools. This information helps you in troubleshooting Versant issues.
Operation	Requirement service method name.
Additional Info	Some additional debug information.

Requirements service transaction monitoring is always enabled. It helps you in identifying transactions running for a long time on Architect/Requirements. Use your web browser to print or refresh the snapshot.

When troubleshooting a performance related issue on your Architect/Requirements instance, you can use this feature in conjunction with requirements service transaction logging. Once the transactions are complete, they are not displayed in the requirements service transaction monitoring report. However, you can refer the transaction start and end records in the requirements service transaction log.

Appendix A: Glossary

This appendix defines Architect/Requirements terms.

A

Access Control

Protection of objects from viewing, modification, or deletion by unauthorized users. Access control is enforced through user access privileges and security profiles. See also *Access Privilege*, *Maximum Privilege Level*, and *Security Profile*.

Access Privilege

User's level of access to a specific project. The access privilege for any project cannot be higher than the user's maximum privilege level. The access privilege must be equal to or lower than the maximum privilege level. A user can be assigned a different access privilege for each project. Compare with *Maximum Privilege Level*.

C

Circular Reference

Defining trace link from a complying object back to its defining object. Circular references are allowed only if the defining and complying trace links are of different subtypes. See also *Complying Object*, *Defining Object*, and *Trace Link*.

Creator

Security profile identifier for the user who created the object to which the security profile applies. This general identifier can be used instead of a specific user name in assigning a permission to the object's creator. Compare with *Everyone* and *No One*. See also *Security Profile*.

E

Enterprise Administrator

Access privilege with which the user can perform any action on all objects in all projects. Compare with *Project Administrator*. See also *Access Privilege* and *Maximum Privilege Level*.

Everyone

Security profile identifier representing all Architect/Requirements users. For the object to which the security profile applies, this general identifier can be used to assign a permission globally to all users. Compare with *Creator* and *No One*. See also *Security Profile*.

F

Full Control

Security profile rule that allows the user to view, modify, and delete the object to which the security profile applies. **Full** permission also allows the user to modify the security profile itself. Compare with *Modify Permission* and *Read Permission*.

M

Maximum Privilege Level

Highest access privilege that can be granted to a specific user for any project in Architect/Requirements. The user's maximum privilege level applies to all projects, whether or not the user currently has access to any project. Compare with *Access Privilege*.

Modify Permission

Security profile rule that allows the user to view and modify the object to which the security profile applies. **Modify** permission prevents the user from deleting the object and from modifying the security profile itself. Compare with *Full Permission* and *Read Permission.*

N

No Access Privilege

Access privilege denying the user access to a specific project. For that user name, the project does not appear in the Systems Engineering and Requirements Management module. Compare with *Read and Write Privilege* and *Read Only Privilege*. See also *Access Privilege* and *Maximum Privilege Level*.

No One

Security profile identifier denying a permission to all Architect/Requirements users. No user has the specified permission for the object to which the security profile applies. Compare with *Creator* and *Everyone*. See also *Security Profile*.

P

Permission

Rights that determine the actions that a user can perform on individual objects. Compare with *Privilege*.

Privilege

Rights that determine a user's access to specific projects. Compare with *Permission*.

Project Administrator

Access privilege with which the user can perform any action on all objects in a specific project. Compare with *Enterprise Administrator*. See also *Access Privilege* and *Maximum Privilege Level*.

Property Definition

Set of properties that apply to all objects of a given type. Certain property definitions are automatically assigned to each object type. These system-defined property definitions always apply and cannot be modified. A project administrator can extend the system-defined properties by creating custom property definitions and applying those properties to object types. See also *Subtype* and *Type Definition*.

R

Read and Write Privilege

Access privilege with which the user can view and modify, but not delete, the objects in a specific project. Compare with *No Access Privilege* and *Read Only Privilege*. See also *Access Privilege* and *Maximum Privilege Level*.

Read Only Privilege

Access privilege with which the user can only view, not modify, the objects in a specific project. Compare with *No Access Privilege* and *Read and Write Privilege*. See also *Access Privilege* and *Maximum Privilege Level*.

Read Permission

Security profile rule that allows the user to only view the object to which the security profile applies. A user with **Read** permission can attach notes to the object and can create trace links to the object. However, the user cannot modify or delete the object, and cannot modify the security profile itself. Compare with *Full Permission* and *Modify Permission*.

S

Security Profile

Set of access rules that control user actions on individual objects. A security profile defines a specific set of users, and specifies each user's permission for the objects to which the security profile applies. A given security profile can be applied to any number of objects. See also *Creator*, *Everyone*, *Full Permission*, *Modify Permission*, *No One*, and *Read Permission*.

Subtype

Custom object type definition based on system-defined object type definition. Project administrators can create subtypes of folders, requirements, and notes. Each subtype inherits the system-defined properties of the built-in object type on which the subtype is based. In addition, project administrators can apply user-defined properties to each subtype. Users can assign subtypes when creating new objects in the Systems Engineering and Requirements Management module. Users can also assign subtypes by changing the **Subtype** property of existing objects. See also *Property Definition* and *Type Definition*.

T

Type Definition

Stores the list of properties that apply to all objects of a given object type. Each project has its own set of type definitions, so the customization is done on a project by project basis. Project administrators can customize object types by editing the properties that apply to the type definition. See also *Property Definition* and *Subtype*.

U

User

Object that represents an individual who is registered in the Architect/Requirements database. Licensing and access to projects are determined by each user's access privilege. See also *User Profile*.

User Access

See *Access Privilege*.

User Profile

Set of properties that record information about a user. Those properties are **Maximum Privilege**, **User Access**, **Security Profile**, **User Name**, **First Name**, and **Last Name**. See also *User*.

Appendix B: System-Defined Properties in the Administration Module

This appendix describes the system-defined properties of the object types used in the administration of a project.

Overview of System-Defined Properties

Architect/Requirements automatically assigns system-defined properties to all object types in both modules. In the Administration module, the object types are:

- Property definitions
- Type definitions
- Reports, templates, and style sheets
- Users and user groups
- Security profiles
- Activators

System-defined properties fall into the following categories:

- Read-only, with a value that cannot be changed.
- Editable, with a default value that can be changed by a project administrator.

Table of System-Defined Properties

In the Administration module, all viewable system-defined properties that apply to the selected object are displayed in the properties table, below the content table.

Table B-1 lists each system-defined property, including its application and description.

Attempting to remove system-defined properties from the applicable schema objects may have unintended consequences.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
Activators	Type Definition	Object that triggers an action. This property is editable.
Additional Privilege	Access Link, User	Additional privilege assigned to a user or required for a project. Values are **Script Authoring**, **Architect**, and **DFSS Package**. This property is editable.
Apply To New Descendents	Security Profile	Determines whether the security profile is inherited by new members of the object to which the security profile is applied. Values are the following: **MEMBER** All new members of the object inherit this security profile. **None** New members of the object do not inherit a security profile. This property is editable.
Change Approvers	Change Approval, Security Profile	List of change approvers. This property is editable.
Change Log	Type Definition	Captures the history of modifications to an object of a given type. This property is editable.
Change Notifiers	Change Approval, Security Profile	List of change notifiers. This property is editable.
Change Time	Type Definition, Property Definition, Security Profile	Date and time when the object was last modified. This property is read-only. Change events depend on the object type: • For a type definition, the value is updated when a subtype is created as a child of the type definition.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
		• For a property definition, the value is updated when the property definition is added to a type definition. • For a security profile, the value is updated when a user or a user group is added to or removed from the security profile. For more information, see Updates to Change Time and Change User Properties in the chapter ***Unsatisfied xref reference**, **Unsatisfied xref title***.
Change User	Type Definition, Property Definition, Security Profile	Login name of the user who last modified the object. Change events for this property are the same as those described above for the Change Time property. This property is read-only. For more information, see Updates to Change Time and Change User Properties in the chapter ***Unsatisfied xref reference**, **Unsatisfied xref title***.
Choice List	Choice Definition	List of choices for this choice property. This property is editable.
Complying Objects Count	All	Number of complying objects. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.
Content	Diagram, Spreadsheet	Binary content of the object. This property is editable. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates.
Content Type	Text property definition	Identifies whether the text property value is a plain string or a URL. If it is a URL, the value is presented to the user as a hyperlink. This property is editable.
Create Time	All	Date and time when the object was created. This property is read-only.
Create User	All	Login name of the user who created the object. This property is read-only.
Current Access	Access Link	Choice attribute object describing the access this user has to the project being accessed. This property is editable.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
Current Value	All property definitions	Default value of a property definition. This property is editable.
Database Class Name	Type Definition	Versant class name for objects of this type. This property is read-only.
Date Format	Date Definition	Valid format for this property value. This property is editable.
Default View	Folder Type Definition	Named view assigned by default to instances of this folder type definition in the Systems Engineering and Requirements Management module. When a folder is selected in the navigation tree, the view determines the column settings in the content table. Users can override the default view and can select other views. This property is editable.
Defining Objects Count	All	Number of defining objects. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.
Description	Search, objects in other Teamcenter applications	Text describing an external object (in another Teamcenter application) that is referenced by an object in Architect/Requirements. This property is editable.
Document Template Rules	Document Template	Controls display of full content reference links as hyperlinks in the **Preview** tab and window. This property is editable.
Effectivity Date	User	Date as of which to display versions for this user. Versions created from this date to the present are displayed. This property is editable. This property automatically reflects the user's latest settings in the **Effectivity** field in the Systems Engineering and Requirements Management module. Conversely, editing this property in the Administration module changes the user's settings in the **Effectivity** field.
Effectivity Label	User	Name of the baseline to display for this user. Versions assigned to this baseline are displayed. This property is editable. This property automatically reflects the user's latest settings in the **Effectivity** field in the Systems

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
		Engineering and Requirements Management module. Conversely, editing this property in the Administration module changes the user's settings in the **Effectivity** field.
Effectivity Type	User	Effectivity rule under which objects are displayed for this user. Values are **Current Frozen Version**, **Current Version**, **Frozen As Of Date**, **Version As Of Date**, **Baseline**, and **Baseline with in-Work**. This property is editable. This property automatically reflects the user's latest settings in the **Effectivity** list in the Systems Engineering and Requirements Management module. Conversely, editing this property in the Administration module changes the user's settings in the **Effectivity** list.
Email	User	User's E-mail address. This property is editable.
Error String	Excel Template	Text string placed in the property columns of the Excel export file if properties in an Excel template do not exist in the database. The default string is *Property does not exist*. This property is editable.
Events	Activator, Macro	Identifies which events cause the activator or macro to run, for example, **After Modify** or **Before Delete**. This property is editable.
Excel Template Rules	Excel Template	Controls the placement of data in a Microsoft Excel export file. With the **Apply Packing** value, data for two or more objects is placed on the same row. This property is editable.
First Name	User	User's first name. This property is editable.
Form Values	Activator, Macro	List of properties for which the user is prompted when running an activator or a macro. This property is editable.
Format	Numeric Definition	Valid format for this property value. This property is editable.
Formula	Numeric Definition	Allows the property value to be calculated based on subordinate objects. Values are **Average**, **Maximum**, **Maximum**, **Multiply**, and **Sum**. This property is editable.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
Full Control	Security Profile	List of users who can perform any action on objects to which this security profile applies. This property is editable.
Has Trace Links	All	Identifies whether the object has trace links or not. **True** indicates that the object has trace links, while **False** indicates that there are no trace links for the object.
HTML	Requirement Type Definition, Note Type Definition	Full content of a requirement, paragraph, or note, including graphics and tables. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates.
Icon	Type Definition	Graphic identifying objects of this type. The default icon for a requirement, for example, is a puzzle piece. This property is editable.
Input Required	Choice Definition	Determines if the property in the Systems Engineering module can have a blank value or if at least one choice is required. This property is editable. The **No** checkbox is checked by default, which allows the choice property value to be blank and allows users to clear existing choices. When the **Yes** checkbox is checked, the value of each occurrence of the choice property is retained. The next time each value is edited, at least one choice is required. Otherwise, the previous value is retained and an error message is displayed.
Instance Security Profile	All	Security profile assigned to an instance of an object. This property is editable.
Last Name	User	User's last name. This property is editable.
Launcher URL	All	Object URL used to launch the Architect/Requirements client when navigating to the object. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.
LightWeight Properties	Individual objects selected in the Systems Engineering and Requirements Management module	Text property that can be created for any object and can be viewed in the Architect/Requirements client. The flexibility is useful for objects that are used in interfaces with other Teamcenter products. This property is editable only through the Architect/Requirements API.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
		For more information about using Tcl or the Java API, see the *Systems Architect/Requirements Management API Reference* manual.
Maximum Privilege	User	Maximum access privilege that this user can have to any project. This property is editable.
Maximum Privilege Level	User	Maximum access privilege that this user is allowed in the current project. This property is editable.
MHTML	Requirement Type Definition, Note Type Definition	Full content of a requirement, paragraph, or note, including graphics and tables. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates.
Menu Text	Menu Folder, Menu Item	Text of the menu or menu item as it appears in the Architect/Requirements client. If this value is blank, the name of the object is used as the text. This property is editable.
Mnemonic	Menu Folder, Menu Item	Mnemonic used on the menu or menu item. This property is editable.
Modify And Read Access	Security Profile	List of users who can modify objects to which this security profile applies. This property is editable.
Module	Menu Folder	Sets the corresponding menu for display in any combination of the Systems Engineering and Requirements Management module, the Administration module, and the Search module. This property is editable.
Multiple Choice	Choice Definition	**True** if this is a multiple-choice property definition. **False** if this is a single-choice property definition. This property is editable.
Object Template	Type Definition	Determines the data that is exported to Microsoft Word for objects of this type definition. This property is editable.
Open Icon	Type Definition	Graphic identifying currently open objects of this type. This property is editable.
Open Property	Activator, Macro, Text Definition, Where Clause	Determines which property the Architect/Requirements client uses when an **Open** command is run. For example, when an activator is opened, the **Script**

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
		property is displayed. When a text property is opened, the **Current Value** is displayed. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates.
Output Template	Search	Saved view, Word document template, Excel template, or live Visio stencil used to display the output from a Search. This property is editable.
Output Type	Search	Type of output format for a report. Values are **Window**, **Word**, **Excel**, **Visio**, and **None**. This property is read-only.
Override Object Template	Type Definition	Object template that overrides the default object template for an object. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates.
Owner	All	LOID number of the containing object. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.
Packages	Project	Packages available in the project. Packages correspond to this property's valid values, which are **Change Log** and **Versions**. This property is editable.
Pass Owner to TcL Context	Choice Definition	Determines whether a **Picklist** activator applied to the choice definition must recognize the ID of the object to which the corresponding choice property is applied. Values are **Yes** and **No**. This property is editable.
Path	All	Sequence of LOIDs from the project node to the object. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.
Projects	User	Projects to which this user has access. This property is editable.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
Properties	Type Definition	List of properties that apply to objects of this type. This property is editable.
Property	Reference Link	Name of the property being referenced. This property is read-only.
Read Access	Security Profile	List of users who can only view objects to which this security profile applies. Users with higher permission automatically receive this permission. This property is editable.
ROIN Counter	Requirement Type Definition	Used to assign custom prefixes and formatting to the ROINs of requirement subtypes. This property is editable.
Routing Status	Requirement	Used by Teamcenter Community to track the stages of approval for requirements. This text property must be applied to requirement type definitions by the Architect/Requirements project administrator. This property is editable.
Script	Activator, Macro, Search, Where Clause	Prewritten group of commands. May be written using Tcl, Java or C#. This property is editable.
Script Type	Activator, Macro, Where Clause	Type of script, such as Tcl, Java, or C#. This property is read-only.
Shared State	Search, View, Menu Folder, Menu Item	User visibility of Search module reports, content table views, and custom menus and menu items. Values are the following: **Private** Visible only to the creator. Not displayed for any other user. **Private** is the default value for each new report, view, menu folder, and menu item. **Private** also allows the user to disable the object temporarily, for example, during editing. **Pending** Marked as a request to be made public by a project administrator. The creator can set the value to **Pending**. **Public**

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
		Visible to all users in the project. Any user who has **Project Administrator** privilege can set the value to **Public**. This property is editable.
Show Non-Effective	Trace links	Setting to make the non-effective object visible in the **Links** tab for trace links.
Snapshot	Diagram	Image of the diagram. This property is read-only. This is a non-displayed property for internal use.
Snapshot Filename	Diagram	File name of the diagram image. This property is read-only. This is a non-displayed property for internal use.
Style Sheet	Document Template	List of Microsoft Word styles associated with a document template. The style sheet determines the formatting of data that is exported to a Word document based on the document template. This property is editable.
Style Sheet Source	Project	File name of the Word document from which the style sheet was imported. This property is editable. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates.
Time Flag	Date Definition	Specifies whether the date includes the time of day. This property is editable.
Time Format	Date Definition	Valid format for the time of day. This property is editable.
Type	Type Definition	Built-in object type on which this subtype is based. This property is read-only.
Type Name	All	Type of schema object. This property is read-only.
Update Choice List Dynamically	Choice Definition	Used to assign a **Picklist** activator to populate the choice list automatically. The activator script, written in Tcl, returns a result that updates the choice list dynamically in the Systems Engineering and Requirements Management module. This property is editable.

Table B-1. System-Defined Properties in Administration Module

Property	Applies To	Description
Usage	Activator, Macro, Where Clause	Description or explanation of the object, for example, its purpose or how it is used. This property is editable.
User Access	Project	Access privilege to the selected project for the current user of this session. This property is read-only.
User Access	User	Access privilege to the project for the user selected in the **Users** folder. This property is editable.
User Access Level	Project	Numeric equivalent of the User Access property. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates.
Word URL	Requirement Type Definition, Note Type Definition	URL for editing the object's content in Microsoft Word. The URL can also be pasted into the **Address** bar in Microsoft Internet Explorer and edited in the browser. This property is read-only. This is a non-displayed property. The name can be entered in tags for object templates and Excel templates, and also can be used in Search criteria.

Appendix C: Date Style Generator Utility

Architect/Requirements provides OOTB date style sheets for all supported locales to export data in Excel using views. You can extend the support for these styles by adding locale-specific styles (**.properties** files) in a custom folder specified in the **Custom.Folder.Path** Web application configuration parameter. You can use the **Date Style Generator** utility to generate these custom date properties file.

The following are the prerequisites for the utility:

- JDK installed on the user's machine. The **JDK_BIN** environment variable should be set to a valid *JDK/bin* home folder.
- Microsoft Excel 2013 or Excel 2016 to generate the **.properties** file.

To generate a custom date style sheet using the utility:

1. Extract the **DateStyleCreator.zip** file to a temporary folder (for example, **C:\TcSE_DateStyle_Generator**).
2. In the extracted contents, the **templates** folder includes the **DateFormat_Template.xlsx** file, which contains the mapping information between the Architect/Requirements date formats and the Excel date styles.
3. Take a backup of **DateFormat_Template.xlsx** for future use.
4. Set the regional setting of the machine to the language for which you are creating a style.
5. Open **DateFormat_Template.xlsx** in Excel.
6. The values in the first two columns (**TcSE Date Format** and **TcSE Time Format**) represent all possible combinations of Architect/Requirements date and time formats.

   The **Display Date** column represents the Excel date format corresponding to Architect/Requirements date and time format. For example, the **Long** date format in Architect/Requirements is represented by applying custom date formatting in Excel as **mmmm d, yyyy**.

   For a given locale, modify the formatting on every cell of **Display Date** column, such that the date format matches with the Architect/Requirements date format for the given locale. To apply the date format, right-click a cell and select **Format Cells**. The **Custom** category can be used to format the cell.

   Save and close the file after modifying it.
7. Run **dateStyleCreator.bat** from the utility folder. It prompts for the location of the template file. Enter the full path of the **.xlsx** file modified in step *6* (for example, **C:\TcSE_DateStyle_Generator\templates\DateStyle_Format.xlsx**).

8. The utility creates the property file (for example, **DateStyle_en_GB.properties**) in the **templates** folder.
9. Copy this file to the folder specified in the **Custom.Folder.Path** Web application configuration parameter.

Appendix D: Project Package Property

Architect/Requirements provides the following package options for use as the **Package** property of a project:

- **Version**

  The majority of product development involves changing and improving an existing product, rather than starting from scratch. As a result, the product structures, requirements, and functions remain essentially the same as in the original design, but change slightly as the product matures. To accommodate this evolutionary process, Architect/Requirements employs versions and variants for requirements and building blocks.

  Each version is an independent object, with its own relationships to other objects, including child objects and defining and complying objects.

  For information on **Version**, see *Enabling the Versions Package for the Project*.

- **Change Log**

  A change log captures the history of modifications to an object of a given type definition.

  For information on **Change Log**, see *Change Logs*.

- **Diagramming**

  Projects are created with three default stencils for the three basic diagram types:

  - Ported
  - Un-ported
  - Static tree

  Installing the diagramming package creates additional stencils for some common diagram types like UML, IDEF0, flow chart, ERD, block diagram, and data control flow.

- **Standard Reports**

  Architect/Requirements provides built-in reports through the optional Standard Reports package.

  For information on **Standard Reports**, see *Reports*.

- **Synergy**

  Installing the Synergy package updates the project's schema to support the Synergy interface.

  For details on Synergy interface, see the chapter *Controlling Requirement Content Changes Through Telelogic Synergy* in the *Systems Architect/Requirements Management User's Manual*.

- **Power Users**

  Power users are non-administrative users who are assigned with project-specific privileges to assist with the project administration.

  For information on **Power Users**, see *Power Users*.

You must have the **Project Administrator** privilege for the project to modify the Package property of a project.

To modify the Package property of a project:

1. In the navigation tree, select the **Projects** node.

   The content table displays all projects to which you have access.

2. Select the project in the content table.

   The **Properties** tab or window displays all viewable properties for the project.

3. In the **Value** column, double-click the **Packages** property value.

   The Multi-Choice dialog window is displayed.

4. Select the packages that you want to apply and click **OK**.

Appendix E: Controlling Requirement Content Changes Through IBM Synergy

Overview of the Architect/Requirements Interface With Synergy

IBM Synergy product support is provided by IBM.

The Architect/Requirements interface with IBM Synergy™ 6.5 brings together primary capabilities of both applications:

- Create requirements in Architect/Requirements, and enter content in the database using Microsoft Office Word.
- Add requirements to Synergy and use its check in and check out functions, accessible in Architect/Requirements, to manage content changes. A Synergy session is not required for these actions.

  A checked-in requirement is released by the check in user and is frozen in Architect/Requirements. It is unfrozen automatically when it is checked out.

  A checked-out requirement becomes a new version in Architect/Requirements.

Each application maintains references for synchronization with the corresponding objects in the other application:

- In Architect/Requirements, the **SynergyID** property shows the full path to a requirement in a Synergy project work area.
- In Synergy, the **TcSE_URL** property shows an object's URL in Architect/Requirements.

Architect/Requirements remains the authoring and editing tool. Do not enter content in Synergy. Otherwise, synchronization problems may occur.

From Synergy, you can open a requirement object and view the content in a browser window. This content automatically includes a hyperlink, named **Goto**, which you can click to navigate to the requirement in Architect/Requirements. The Architect/Requirements client is not required for viewing in Synergy.

The project administrator enables the interface by adding the **Synergy** package, together with the **Version** package, to the Architect/Requirements project in the Administration module.

Synergy Package Additions to Architect/Requirements Project Schema

The **Synergy** package updates the project schema as follows:

- Adds the following subfolders to the **Activators** folder:

Synergy	Defines the Synergy custom menu in the Systems Engineering and Requirements Management module and contains a menu item for each menu command. This subfolder has the **Menu Folder** subtype.
Synergy Macros	Contains activators and macros that carry out various Synergy interface functions. This subfolder has the **Admin Folder** subtype.

  For more information about using activators, see the *Systems Architect/Requirements Management API Reference* manual.

- Adds the following properties to the **Property Definitions** folder:

Synergy Check-In Comment	Plain text information entered the last time the requirement was checked in to Synergy. The default comment is **Check In from TcSE**.
SynergyID	Full path to a requirement in a Synergy work area.
SynergyTask	Synergy task to which a checked-out requirement is assigned.

  The Synergy properties are automatically applied to the **Requirement** type definition in the **Type Definitions** folder. Subsequent new subtypes based on the **Requirement** type definition inherit the Synergy properties. Before existing subtypes can be used with Synergy, these properties must be manually applied to the subtypes, including the **Paragraph** subtype. For more information about modifying the properties of a type definition, see the *Systems Architect/Requirements Management Project Administrator's Manual*.

- Adds the following templates to the **Reports and Formatting** folder:

Synergy Document Template	o Applies the formatting specified in the **Default Style Sheet** to the Synergy object content. o Assigns the **Synergy Object Template** to the **Requirement** type definition. The project administrator is responsible for modifying the **Synergy Object Template** assignment to include any subtypes to be added to Synergy.

Synergy Object Template	Contains tags that specify the content that is inserted in the Synergy object. This data includes the **Goto** hyperlink in the Synergy object content.

For more information about templates, see the *Systems Architect/Requirements Management Project Administrator's Manual*.

Enabling the Synergy Package for an Architect/Requirements Project

The Architect/Requirements **Synergy** package contains the components of the interface with Synergy. The project administrator installs these components by adding the **Synergy** value to the **Packages** property for the project. For more information, see Synergy Package Additions to Architect/Requirements Project Schema, earlier in this chapter.

- The **Synergy** package cannot be removed from the project after it is added.
- Synchronization problems may occur if requirement content is entered directly in Synergy. Use only Architect/Requirements to author and edit the content of requirements that are added to Synergy.

- You must have **Project Administrator** privilege for the project.
- The **Version** package also must be added to the project. The **Version** package cannot be removed from the project after it is added.

1. On the module bar, click the **Administration** button to access the Administration module.
2. Select the project node in the navigation tree or the content table.

   The **Properties** tab or window displays all viewable properties for the project.
3.

   In the **Value** column, double-click the value for the **Packages** property.

   The Multi-Choice dialog window is displayed.
4. Check the **Synergy** check box, and then click **OK** to close the dialog window.

   If the **Version** check box is cleared, check it before you click **OK**.

   The **Synergy** and **Version** packages cannot be removed if you continue this procedure.

5. Click **Yes**.

   This action cannot be reversed.

   The **Synergy** value is added to the **Packages** property and the Synergy package is enabled.

Adding Requirements to a Synergy Project

Requirements must be added to a Synergy project before their content can be managed through the check in and check out functions. A Synergy session is not required for this procedure.

- If a folder is selected, only its top-level requirements are added. Its nested folders, if any, are also selected and only their top-level requirements are added. Children of top-level requirements are not added.
- If a selected folder contains objects other than requirements and folders, those objects are not added.
- For parent requirements selected individually, direct children and lower-level descendants are not added.
- Requirement hierarchies in Architect/Requirements are not preserved in Synergy. They reside at the same level in the Synergy directory.
- The Synergy properties must be applied to the type definition of each requirement. For more information, see Synergy Package Additions to Architect/Requirements Project Schema, earlier in this chapter.
- You cannot add requirements that are frozen.

1. Select the requirements, pull down the **Synergy** menu, and choose **Migrate**.

   The Synergy Login dialog window is displayed if you are not logged in to Synergy. Enter your Synergy user ID and password, and then click **OK**.

   The Select a Synergy Project dialog window is displayed.

2. Click the Synergy project, and then click **OK**.

   The Select Folder for Files dialog window is displayed.

3. Navigate to and select the Synergy directory where you want to add the requirements, and then click **OK**.

   At the bottom of the main window, the status bar displays a progress indicator. Then, an information message states that the migration is complete.

   For each selected requirement:

   - A check mark appears on the object type indicator, showing that the requirement is checked out to your Architect/Requirements user name.
   - The **SynergyID** property shows the full path in the Synergy project.
   - The **Synergy Check-In Comment** and **SynergyTask** values are blank.

Procedure Notes

Step 1: You can select individual requirements in the content table or the **Links**, **Where Used**, or **Versions** tabs or windows. You can select a folder in the navigation tree, the content table, or the **Links** tab or window.

Checking In Requirements to Synergy

A check mark is shown on the object type indicators of requirements that are checked out from Synergy. When changes are complete, check in the requirements to make them available to other users. A Synergy session is not required for this procedure.

You can select a folder the check in its top-level requirements simultaneously. Only the folder's top-level requirements are checked in. Its nested folders, if any, are also selected and only their top-level requirements are checked in. Children of top-level requirements are not checked in.

For parent requirements selected individually, direct children and lower-level descendants are not checked in.

- If a selected folder contains objects other than requirements and folders, those objects are not checked in.
- This procedure freezes the requirements. You cannot check in requirements that are already frozen.

1. Select the requirements, pull down the **Synergy** menu, and choose **Check-In**.

   The Synergy Login dialog window is displayed if you are not logged in to Synergy. Enter your Synergy user ID and password, and then click **OK**.

   The Check-In dialog window is displayed.

   You can enter a comment by double-clicking the **Values** cell to open a text field. A default comment is entered if you leave this field blank.

2. Click **OK** to start the check in.

   At the bottom of the main window, the status bar displays a progress indicator during this process. Then, an information message states that the check in is complete.

   For each selected requirement:

   - A lock symbol replaces the check mark on the object type indicator, showing that the requirement is frozen.
   - The **Synergy Check-In Comment** property displays your comment or the default comment.
   - The **SynergyTask** property is cleared of any previous value.

Procedure Notes

Step 1: You can select individual requirements in the content table or the **Links**, **Where Used**, or **Versions** tabs or windows. You can select a folder in the navigation tree, the content table, or the **Links** tab or window.

Checking Out Requirements From Synergy

A lock symbol is shown on the object type indicators of requirements that are checked in to Synergy. The lock symbol shows that the requirements are frozen. Check out the requirements to create a new version in Architect/Requirements and make changes to the content. A Synergy session is not required for this procedure.

You can select a folder to check out its top-level requirements simultaneously. Only top-level requirements are checked out; children are not checked out. Any nested folders are selected and only their top-level requirements are checked out.

For parent requirements selected individually, direct children and lower-level descendants are not checked out.

Use only Architect/Requirements to change the content of requirements that are checked out from Synergy. Do not edit such requirements directly in Synergy. Otherwise, synchronization problems may occur.

- If a selected folder contains objects other than requirements and folders, those objects are not checked out.
- The requirements must be frozen. This procedure unfreezes the requirements automatically.

1. Select the requirements, pull down the **Synergy** menu, and choose **Check-Out**.

   The Synergy Login dialog window is displayed if you are not logged in to Synergy. Enter your Synergy user ID and password, and then click **OK**.

   The Select a Synergy Task dialog window is displayed.

2. Click the Synergy task to which you want to assign the check out, and then click **OK** to start the check -out.

   At the bottom of the main window, the status bar displays a progress indicator. When the check out is complete, an information message is displayed.

   For each selected requirement:

   - A new version is created in Architect/Requirements.

     You can access all of the requirement's versions through the **Versions** tab and window. In the content table, you can use the **Effectivity** list to access versions by category. For more information, see ***Unsatisfied xref title*** in chapter **Unsatisfied xref number**, ***Unsatisfied xref title***; and ***Unsatisfied xref title*** in chapter **Unsatisfied xref number**, ***Unsatisfied xref title***.

   - A check mark replaces the lock symbol on the object type indicator, showing that the requirement is unfrozen and checked out.
   - The **SynergyTask** property shows the assigned task.

Procedure Notes

Step 1: You can select individual requirements in the content table or the **Links**, **Where Used**, or **Versions** tabs or windows. You can select a folder in the navigation tree, the content table, or the **Links** tab or window.

Synergy Interface Customization

The Synergy Interface Toolkit provides a mechanism to link Architect/Requirements requirement objects with Synergy CM. This linking allows Architect/Requirements to remain the content authoring tool, while Synergy CM is used as the configuration management tool. It is a toolkit in that it provides the necessary tools to allow the user to customize the Synergy interface and to create totally new applications and interfaces.

When the Synergy package is installed, several menu items are included, plus several activators and macros. Along with the delivered Synergy Interface Java code, these are the tools to be used in the Customization Toolkit. With these tools, custom behavior and custom applications can be developed.

Notable features added to support the Synergy interface are the createAction RunJava mechanism, the client interface ClientJavaAPI.runMacro() mechanism and the ability to customize icons based on a property being set on an object.

Program Flow

A typical flow of events would be to create a requirement in TcSE, then migrate it to Synergy and check it in. After it has been checked in to Synergy, the user must perform a check out operation in order to create a new version of the object in TcSE to make changes.

The Synergy Interface provides the ability to migrate requirements to Synergy, and to perform Check-In and Check-Out operations on the migrated requirements. These behaviors are accomplished by using customizable Tcl code plus customizable external Java code.

Initially, a Tcl macro is invoked on the TcSE client, which in turn executes external Java code, which then in turn calls the runMacro capability in TcSE to perform further TcSE behavior via Tcl.

A Synergy menu item is selected, which executes Tcl code. This code gathers the selected objects, performs necessary verification functions, gathers needed TcSE information, and then passes information along to the external Java code via the createAction RunJava mechanism, which is documented elsewhere.

The external Java code then verifies input parameters, logs into the Synergy application, performs Synergy queries, prompts the user for input, performs Synergy specific actions such as logging in and creating objects, gathers needed information, and then passes information back to TcSE via the ClientJavaAPI's runMacro() method, which runs a macro in TcSE.

The macro then completes the action by using the information passed to set properties, create versions, and set object status in TcSE, and then displays a completion message dialog.

Once the requirement objects are linked to Synergy, then in the interests of synchronization, certain user actions are discouraged via activators. These include deleting the requirement, manually freezing or unfreezing the requirement, manually modifying the Synergy properties and version creation.

Customization Points

The interface is customizable and customizations can be implemented in Java code or Tcl code. The user can also add new menu commands. Changes can be made to the Synergy menu item Tcl code, to the external Java code and to the finishing macros. The behavior can be changed and modified to more exactly match other processes or to change the behavior entirely, or to tweak the behavior a little.

New Applications and Interfaces

The Synergy interface can be used as a starting point for other interfaces. New applications and interfaces can be created following the same general processing.

The Synergy interface provides a number of new integration features that will facilitate interacting with TcSE. These include:

- createAction RunJava
- ClientJavaAPI class
    - runMacro() method
    - getTempDir() method
- Icon overlays

The Synergy interface can be modified, or it can be copied and changed, or it can be used as a learning tool to understand how the interoperability works.

Users can create custom Tcl code and custom Java code that will be run "on the client" in the client JVM. An example of this might be interfacing with other applications or creating custom interfaces, such as creating a Java form for user input. Also, the mechanism is put in place via the createAction RunJava to allow the external program to interface through the C# interface.

createAction RunJava

The createAction RunJava command is covered elsewhere in the documentation. Its purpose is to allow external Java code to be run from within TcSE via Tcl code. It allows identification of Java classes and execution of methods in those classes from the Tcl code of activators, macros or custom menu items. The location of the .jar file must be identified to TcSE via the Package.Location parameter set by the administrator. The method signatures must follow a prescribed format. The .jar file must always contain a method named connectTcSE with a prescribed signature.

Examples of the usage of the createAction RunJava command can be found in the Synergy interface.

ClientJavaAPI Class

The ClientJavaAPI class is the beginning of a client interface.

In order to call a TcSE macro from the external Java code, the class file must import the ClientJavaAPI class, which can be found in the tcrPackages.jar file. So the tcrPackages.jar file can be added to the classpath, or it can be unzipped and the ClientJavaAPI.class file can be pulled out to the development area.

runMacro() Method

The mechanism to call a macro from external Java code uses one of the ClientJavaAPI class's runMacro() method.

A basic method will be the one that simply calls the macro by name, passing in a space delimited string of LOIDs for the macro to use as selected objects:

```
public String runMacro(String macroName, String objectIDs)
```

Another method uses the macro name and the macro LOID along with the object IDs. In this case, if the macroID is passed and is not null, then the macroName is not used:

```
public String runMacro(String macroName, String macroID,
String objectIDs)
```

Another well used method is one where values are passed into the macro via the Form mechanism, passing in the form properties and the associated values. Again, if the macroID is passed and is not null, then the macroName is not used:

```
public String runMacro(String macroName, String macroID,
String objectIDs, String[] formProps, String[] formValues)
```

getTempDir() Method

There is a mechanism to get a temporary directory if needed from TcSE. The getTempDir() method returns a string containing the system temporary directory that is used by TcSE.

ClientJavaAPI Usage

The ClientJavaAPI class must be available to the custom package. Then, the ClientJavaAPI must be imported with this statement:

```
import com.edsplm.tc.req.client.shared.api.ClientJavaAPI;
```

and instantiated with a statement similar to this:

```
protected ClientJavaAPI javaAPI = new ClientJavaAPI();
```

before it can be used like this:

```
result = javaAPI.runMacro(macroName, objectIDs);
```

or this:

```
result = javaAPI.runMacro(macroName, null, macroParameters,
formProps, formValues);
```

Icon Overlays

The Synergy interface does, and new customizations may, use the hidden **Icon Overlay** property to provides a mechanism for adding custom overlays to object type indicator icons. If the **Icon Overlay** property value is the LOID of a type definition, the icon that is set for that type definition is used as an overlay to the current icon.

The **Icon Overlay** property applies only to folders, requirements, and building blocks and their subtypes. These object types are containers for other types of objects.

The custom overlay is the first one applied, followed by any possible out of the box overlays (such as subtype, variant, frozen, etc.) All out of the box icons are 16x16 pixels in size. In case the icon sizes are not the same, overlay icons are anchored at the top left of the icon.

If the overlay icon has transparent parts, then the original icon shows through. If the overlay icon is opaque, it appears to completely replace the existing icon.

A new **Icon** type definition has been created as a convenience. It is a non-instantiable type definition used solely for creating subtypes of type definitions for using custom icon overlays.

The hidden **Icon Overlay** property can be set via Tcl in a menu item, an activator, or a macro. For example, the following Tcl code sets the custom overlay icon to the icon identified as the icon for a type definition named **BigRedEx**:

```
# get the LOID of the icon overlay
set name "BigRedEx"

# find type def by iterating over all type defs
foreach def [getList $currentProject OBJECT_TYPE_LIST]
    {if {[getValue $def Name] == $name}
           {set iconOverlay $def
           break
    }
}
# real code would ensure that icon was found before continuing
setValue $selected "Icon Overlay" $iconOverlay
```

Any overlay change is always a change on the actual object (involving the object's **Change Time** and **Change User** properties). Also, the change impacts not just the current user, but the overlay is saved in the database and shared with all users in the project.

Java Development Environment

Java knowledge is needed in order to take full advantage of this capability, and will help get the development environment to be set up properly. Different people use different tools and have different ways of doing things, so do what is necessary in order to set up a development environment.

The hard part is setting it all up, extracting the files to the proper location and getting the compiler set up to compile correctly and have it compiled in the right location so that TcSE can find it properly. Once everything is set up correctly, the rest is straightforward.

It is a good idea to create subclasses of the delivered classes, so that any potential changes made will not be overwritten when a new TcSE deployment is made.

These things must be accomplished:

- Set up development environment, preferably with an IDE (Eclipse is free at www.eclipse.org).
- Classpaths must be set up correctly.
- Packages must be set up correctly.
- Must be able to "jar up" the code for distribution.
- Understand how to extend the delivered class in order to create new customized behavior.

The goal is to create a .jar file that can be distributed, or to replace the **tcrPackages.jar** file. A good starting point would be to unzip the **tcrPackages.zip** file, which contains the two external Java files used in the Synergy Interface into the working directory.

Development Hints and Recommendations

New custom java code should be in its own package.

The Java package does not require a main() method, but is simply a collection of methods.

The methods must be called from TcSE in order to be an extension of the TcSE client.

Unzip the **tcrPackages.zip** file and become familiar with the Java source code for the TcSECMInterface.java and SynergyInterface.java, and how they work together, as well as become familiar with the out of the box Synergy macros. These are the parts that constitute the Synergy interface, and the parts that are the best example of the Customization Toolkit.

SynergySchemaSave

If the Synergy Interface files are being modified for use with a specific Synergy installation, there is a SynergySchemaSave macro delivered with the Synergy package that can be used to repackage the changes for redistribution. If the Synergy package is customized and saved into the tcrPackages.jar file, special care is needed during the next TcSE upgrade in order to preserve the customizations, as they will be lost with a client upgrade.

If a new interface is being created from scratch, then a new macro similar to the SynergySchemaSave macro can be created to package up the schema for redistribution.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X