MICRO FOCUS

Dimensions CM

Dimensions CM for Eclipse Guide

Product version: 14.5.2

Last updated: December 18, 2020

Table of Contents

Chapter 1

Overview of Dimensions for Eclipse Integration

Introduction

The Dimensions for Eclipse integrates the request and version management functionality of Dimensions CM, and optionally the request management features of Solutions Business Manager, directly into the Eclipse workspace environment. The integration:

- Simplifies and automates the identification, management, and processing of requests in the Eclipse design environment.
- Incorporates Dimensions CM version management functionality, with rich support for parallel development practices including optimistic locking, stream-based development, and workspace synchronization.
- Supports Windows, Linux, and Mac OS X, enabling organizations to manage and secure their enterprise assets, whilst coordinating and automating change activities that span multiple platforms.

Main Features

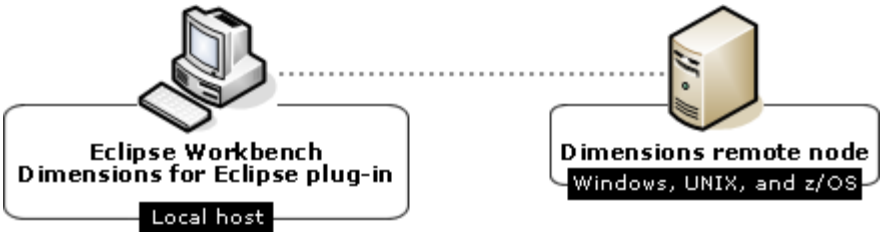
The Dimensions for Eclipse integration enables you to:

- Access Dimensions CM objects directly through the Eclipse workspace. You can access the files directly through Eclipse perspectives such as the Package or Java perspectives, or you can access specific information through the perspectives and views supplied with the integration. See page 17 for details of the different perspectives available in the integration.
- Connect to multiple Dimensions CM servers concurrently through the Dimensions CM Explorer view. See page 13 for details.
- Work with files in a shared work area that is compatible with the desktop and other clients. See page 27.
- Perform request management functions, such as browsing requests, creating requests, actioning requests, and delegating requests, through Dimensions CM Explorer.
- Work on parallel development projects using agile models such as stream-based development, workspace synchronization, and optimistic locking.
- Perform version control activities, such as check in and check out, through package explorer and other Eclipse views.

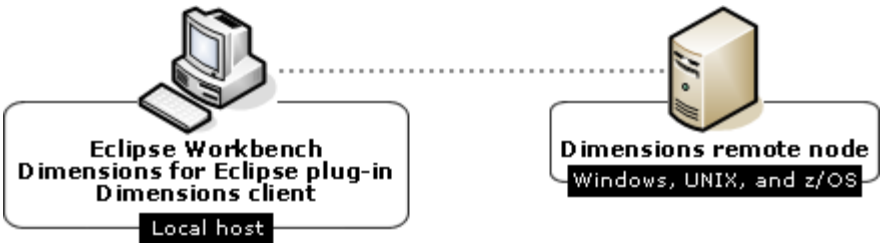
Usage Scenarios

The Dimensions for Eclipse integration gives you flexibility in how you set up your configuration with Dimensions CM. The integration can be used standalone or in junction with the Dimensions CM desktop client, as shown in the following scenarios:

- Connected to a Dimensions CM installation on a remote node with no Dimensions client installed locally.



- Connected to a Dimensions CM installation on a remote node with a Dimensions CM client installed locally.



Dimensions CM Privileges and Role Requirements

Dimensions CM uses privileges and role assignments to define who can perform specific actions within particular products and projects.

To use the functionality in the Dimensions for Eclipse integration, you must have the required privileges and roles in Dimensions CM to perform the required operations.

The table contains some examples of roles or privileges that you need for performing functions.

To...	You need the role or privilege allowing...
Browse or get files	Browsing or getting of the desired items in the project
Check out files	Creation of new revisions of the item types used by files in the project
Add files	Creation of new revisions of the item types used by files in the project.
Add projects	Creation of Dimensions CM Projects and ability to add items/ create directories in the project.

For information on assigning roles and privileges see the *Dimensions CM Process Configuration Guide*.

Support for Rational Developer for System z

Dimensions CM supports Rational Developer for System z (RDz) as an Eclipse IDE. Dimensions CM currently allows control of local projects on the workstation; only files located directly on the workstation can be placed under Dimensions CM control. To handle this, upload rules for Cobol and PL/1 are in the Dimensions CM process model.

There is no direct integration with z/OS projects and mainframe resident data, but RDz itself does include an ISPF client view.

In all other aspects, the Dimensions CM integration to RDz is the same as other Eclipse IDEs.

Chapter 2

Installing the Dimensions for Eclipse Integration

Before You Install

Before you install the Dimensions for Eclipse integration, check the following pre-installation requirements:

- Uninstall any previous versions of the Eclipse integration before you install the current version. On Windows, you can uninstall from **Programs and Features** in the Control panel by selecting **Dimensions for Eclipse**.
- A supported version of Eclipse is installed on the client machine where you are going to install Dimensions for Eclipse.
- You are able to connect to one, or more, of the Dimensions CM servers.
- Your supported version of Eclipse must be run on Java 6 or later.

Upgrading Dimensions Projects

If you are upgrading from versions of Dimensions 10.x, you do not need to perform any migration steps. However, we recommend that you start fresh with all new local workspaces to ensure that your Dimensions metadata is current.

Installing the Integration

You can install the Eclipse integration from an update site hosted by the Dimensions CM server. You can use the same method to install *Appcelerator Titanium Studio* into Eclipse. You can also install Eclipse manually or silently.

For detailed installation instructions see one of the following:

- *Installation Guide for UNIX*
- *Installation Guide for Windows*

Chapter 3

Connecting to Dimensions CM

Introduction

The Dimensions for Eclipse integration allows you to connect to one or multiple Dimensions CM servers. The necessary parameters are saved in a local configuration file, which allows easy access to the same parameters when reconnecting to Dimensions CM.

Enabling Smart Card Authentication

Dimensions supports ActiveIdentity Smart Card service providers. If Smart Card software is installed, you will be able to provide Smart Card authentication information when you open a Dimensions connection.

If the Smart Card software is installed to a different directory from the default location in Dimensions for Eclipse, you must set the correct path in Dimensions for Eclipse.

Also note that the Dimensions integration to Eclipse does not support multiple smart cards. If multiple card readers are enabled and more than one card is active, an error message displays when you attempt to log in.

1. Select Windows | Preferences.
2. Select Team | Dimensions, and enter the correct path in the **Smart Card Library Path** field.

Opening a Connection

You can run multiple open connections concurrently. If a connection times out and is disconnected from a server, it remains disconnected until it is required, and then re-connects automatically.

1. To display the Dimensions CM Explorer, from the Window menu select Perspective | Open Perspective | Other | Dimensions CM.
2. In Dimensions CM Explorer, right-click Dimensions and select **Connect To | *<connection>***.
3. Enter the password for the operating system user that you wish to log-in as, and click the **OK** button.

Creating a New Connection

When you create a new connection, the connection parameters are stored on your local system, which allows you to reopen the connection at a future time without having to re-enter the connection information.

To login obtain the following information from your CM administrator:

- The name of the server.
- (Optional) The port number used to run CM.
- The network protocol used by the server: SDP, HTTP, or HTTPS.
- The database name of the CM product you are working with.
- The connection string for the database.

For security reasons you may be required to login using the HTTP/S network protocol instead of the default Standard Dimensions Protocol (SDP). The Dimensions CM HTTP Connector allows a connection to a server using HTTP/S, for details see the *System Administration Guide*.

1. In Dimensions CM Explorer, right-click the Dimensions node and choose **New | Connection**. The New Dimensions Connection dialog appears. You may also need to enter new connection information when adding Eclipse projects to source control.
2. Enter the name for this connection in the **Connection** field. This is the name that will be displayed in Dimensions CM Explorer after the connection is created.
3. In the **Server** field, the name of the server and optionally the port number to connect to:
   - SDP protocol: `<server name[:port]>`
   - HTTP protocol: `http://<server name[:port]>`
   - HTTPS protocol: `https://<server name[:port]>`
4. In the **DB Name** field, the name of the Dimensions CM database to connect to.
5. In the **CM Connection** field, the database connection (DSN) to connect to on the server.
6. Under **Login Options**, choose whether to log in with user credentials or with a Smart Card.
   - If you will log in with user credentials, enter the user name and password.
   - If you will connect using a Smart Card, select the certificate.
7. Click the **Finish**.

Editing Connection Parameters

1. In Dimensions CM Explorer, right-click on the Dimensions CM node and choose Edit | <*connection*>.

   NOTE: You can only edit a closed connection.

2. Edit the log-in parameters as required.

3. Click OK.

Closing a Connection

1. Right-click the Dimensions connection node in the Dimensions CM Explorer and select Properties.

2. Click the **Disconnect** button.

Deleting a Connection

1. In Dimensions CM Explorer, right-click on the Dimensions CM node and choose Delete | <*connection*>.

   NOTE: You can only delete a closed connection.

2. Click OK to confirm to delete the connection.

Chapter 4

Dimensions CM Explorer

Introduction

The Dimensions for Eclipse integration contains editors, views, and perspectives that are unique to the integration. The main view to access Dimensions CM objects is called Dimensions CM Explorer, which contains a tree view of the projects, streams, and baselines.

Using the Dimensions CM Explorer

Displaying Dimensions CM Explorer

You can display the Dimensions CM Explorer either by launching the view within your current perspective or by using the Dimensions perspective, which is included with the Dimensions for Eclipse integration. The default perspective displays the Dimensions CM Explorer view in the left navigation pane and the Properties view in the pane on the lower-right.

To open the Dimensions CM Explorer perspective:

From the Window menu select Perspective | Open Perspective | Other | Dimensions CM.

To show the Dimensions CM Explorer view within another perspective:

1. Select the area on the workbench to display the Dimensions CM Explorer view.
2. From the workbench **Window** menu, select **Show View** | **Other**.
3. The **Show View** window appears with "other" view choices.
   a. Open the **Dimensions** folder and select **Dimensions CM Explorer**.
   b. Click **OK**.
4. The Dimensions CM Explorer pane opens in the workbench window with the Dimensions CM node collapsed.

   After opening a connection to a server, you can browse a hierarchy of Dimensions CM objects that you are authorized to use. Click on the plus or minus symbols to expand or collapse a node in the hierarchy.

To hide the Dimensions CM Explorer view, click the Close button in the upper right corner of the Explorer pane. This frees workbench window space to display your code or perform other development tasks.

Finding Objects

From Dimensions CM Explorer you can search for requests, projects, streams, baselines, and items.

1 In Dimensions CM Explorer, right-click a node and select **Find | <object>**.

The Selection Wizard dialog displays for the object that you are searching for.

2 Enter your search criteria on the select <object> panel and click **Next** to display the results.

3 On the results panel, select the results to display by checking the box.

4 Click **Finish** to display the results in the appropriate editor.

Invoking Dimensions CM Deployment

Do one of the following:

- In Dimensions CM Explorer right-click a CM connection and select Deployment Views.
- From the Dimensions menu select Deployment Views.

Dimensions CM deployments open in a web browser.

Showing the Dimensions CM Context View for Projects

The Dimensions CM Context view displays the Project Working Branch and Project Working Request for the selected project:

- The **Project Working Request** is the default request that is used to be related *in response to* new item revisions that you create in this project. Note that you can specifically choose a different request than the default request for your items. You can also automatically populate the project working request by first setting a request from the Dimensions CM Explorer view as the *active request*.
- The **Project Working Branch** is the default branch on which new item revisions that you create in this project will be placed.

The Dimensions CM Context view can be enabled in any Eclipse perspective, which allows you to have immediate access to this information while working on your files.

The Dimensions Context view can be set to automatically display when any object file or editor associated with a Dimensions CM project is selected. See the Version Management preferences on page 20 for information on enabling and disabling this preference.

1 Select the area on the workbench to display the Dimensions CM Context view.

2 From the workbench **Window** menu, select **Show View** | **Other**.

3 On the **Show View** window, select Dimensions | Dimensions Context and click OK.

Managing Preferences

Setting General Preferences

You can set preferences for project containers, as well as for the font to use for multi-line properties from the general preference page.

1. Select Window | Preferences
2. Expand the **Team** node, and select **Dimensions**. The general preferences appear.
3. Choose whether to show project containers in the **Dimensions CM Explorer** view, and choose whether to create new projects by default in project containers.
4. Select the **Show Single Eclipse Projects in Explorer** option to display Eclipse projects that are mapped 1:1 to Dimensions CM projects, in the Dimensions CM Explorer view.

If your Smart Card software is installed to a different path from the Dimensions for Eclipse default location, change the Smart Card library path to the correct location.

Setting Console Preferences

You can change the colors that Dimensions for Eclipse uses for the output messages. These messages are usually displayed in the Dimensions CM console view.

1. Select Window | Preferences.
2. Expand the Team | Dimensions node, and select the Console node.
3. Select the **Show Dimensions output in the Console View** option to display Dimensions output in the Eclipse Console view.
4. Select the colors for each of the following console outputs:
   - **Operations** that you performed from the Eclipse interface.
   - **Messages**, such as successful completions, from the operations which you performed.
   - **Error** messages for the failed or problematic operations that you performed.

Setting Preferences for Dimensions CM Request Lists Display

You can change how Dimensions for Eclipse displays requests by modifying your Request Management preferences. This only applies to Dimensions CM requests.

1. Select Window | Preferences
2. Expand the Team | Dimensions node, and select the Request Management node.
3. Choose where to display your Dimensions CM requests lists.

Setting Version Management Preferences

You can change how Dimensions for Eclipse displays items by modifying your Version Management preferences.

1 Select **Window | Preferences.**

2 Expand the **Team | Dimensions** node, and select **Version Management**.

3 Choose from the following:

- **Delete unmanaged resources on replace** to delete all files in your local workspace when you perform a replace.
- **Consider file contents in comparisons** to evaluate the contents of two files if the modification time is identical when performing operations.
- **Attempt to find common ancestor baseline automatically** to have Dimensions CM find the common ancestor for comparisons, instead of having to select it manually.
- **Synchronize workspace with home stream on switching to the foreign stream** synchronizes your workspace when you switch to a foreign stream and warns you if the local workspace is out of sync with the home stream.
- **Preselect project default requests where possible** lets the integration select the default requests for projects when you perform team operations.
- **Clean timestamps during status update** clears the local modification status for any files that have not been changed since they were last retrieved from the repository, but for which the timestamp has been updated (or *touched*).
- **Undo Checkout on checkin of files without changes** automatically undoes check-out of any files that are unchanged, when attempting to check in locked files.
- **Expand substitution variables** expands all substitution variables in source files by default. Avoid using non-reversible substitutions as these will not be removed when you commit changes to the repository.
- **Auto share on import** automatically shares projects imported into Eclipse with Dimensions CM. For more on this use case, please see page 87.

4 Under **Appearance**, choose from the following options:

- Under **Save dirty edits before Dimensions operations**, determine if Dimensions will automatically save the unsaved updates that you have made before performing the Dimensions source control operation. The options are **Never**, **Prompt**, or **Auto save**.
- **Show project and Baseline lists in** lets you choose whether to open project and baseline lists in the editor or in the view (which is usually right-bottom and smaller).
- **Use multipage layout** will cause different lists of projects or baselines to be shown in a single view or editor with tabs at the bottom. If disabled, each has its own view or editor.
- **Automatically start Dimensions Context** opens the Dimensions Context view whenever you select a project or baseline. See page 19 for more information.

- **Maximum number of comments to remember** specifies the number of entries to remember in the **Choose a previously entered comment** drop-down list on the Commit Wizard.

5 Optionally, under **Auto-merge**, choose to have Dimensions for Eclipse attempt to auto-merge all non-conflicting changes when you perform an Update.

6 On the **File Modification** node under **Version Management**, you can set how Eclipse responds when you attempt to edit read-only files:

- Select to *check out to make writeable, make writeable*, or *prompt* when you attempt to modify a read-only file, which is under Dimensions CM control.
- Select to *make writable*, or *prompt* when you attempt to modify a read-only file, which is not under Dimensions CM control.
- Select to *make writable*, or *prompt* when you attempt to modify a read-only file, which is checked out from Dimensions CM.

7 Change your Label Decoration preferences by selecting **Label Decorations** under **Version Management**:

Preference	Action
Show deep status for folders	Shows composite status for the folders. See following note for information about using this setting with the following setting.
Always show deep status for projects	Shows composite status for the children. **NOTE** *Always show deep status for projects* overrides *Show deep status for folders* for project nodes if the latter is disabled and causes projects (but not folders) to display cumulative status.
Show remote info for projects	Shows connection details for a project.
Show base revision for files	Shows the base revision of the file that was fetched in ().
Show remote revisions for files	Shows the file is different from remote revision by stating remote revision number in [].
Show locally modified status as text	Marks locally modified files with a text decoration (>).
Show locally modified status as image	Marks locally modified files by changing the file icon. The icon contains a pen symbol on the top right.
Show local mode	Marks files that are in local mode by changing the file icon. The icon contains a wrench.

8 Select the **Requests** node under **Version Management** to set the following preferences for handling requests:

- **Using work list by default**: Choose whether to display your working list of requests or your Dimensions inbox when checking files out and in.
- **Show project id on request selection page**: This option displays/hides the **Project** column on the Relate Requests page of the Commit Wizard. This column displays the Dimensions project of each resource.

- **Remove from my working list after successful checkin / commit operations:** Select this option to remove requests from your working lists after you have successfully checked in or committed files in response to those requests.

Chapter 5

Working with Projects and Streams

Introduction

About Dimensions Source Control

The Dimensions for Eclipse integration provides access to Dimensions CM source control functionality directly from the Eclipse environment, which means that you can perform source control actions, such as check in and check out, without exiting your Eclipse environment.

You can map Eclipse projects either to Dimensions projects or Dimensions streams. Dimensions for Eclipse also lets you group multiple projects into a *project group*, or organize Eclipse projects into project containers. Project containers are a special type of Dimensions project that may contain more than one IDE project, for details see page 84. The Eclipse project files are placed into the Dimensions CM repository based on the mapping.

When you add your Eclipse projects to the Dimensions CM repository, Dimensions CM stores information about the Eclipse project, such as integration type and project name, in the Dimensions CM repository. See page 26 for details.

About Eclipse Project Data

When you add an Eclipse project to Dimensions CM control, Dimensions stores additional data about the project, such as the IDE type (Eclipse), the actual project name in Eclipse, and other project specific information. This information is stored as attributes for the Dimensions CM project that is associated with the Eclipse project.

This information is stored either as a project marker file with the extension “ecl” (when working with Eclipse containers), or it is stored as attributes for the Dimensions CM project or stream. Additional information is associated with Eclipse Project Containers that allows them to be explicitly identified in the Dimensions CM Explorer.

About Optimistic Locking

Optimistic locking allows you to check in files that were not checked out. You check in the updated work file into the repository directly, or if the repository revision has changed, you merge the local file and the repository revision and then check in the changed file into source control. Optimistic locking differs from the classic "pessimistic" approach, where a user had to explicitly check out the file before being able to place the updates into the repository.

Optimistic locking works well in an Eclipse development environment where developers are working concurrently on projects. A developer may have the entire project locally and need to make changes, but he may not know which files are affected until beginning work on the code. Optimistic locking allows the developer to alter the code, and then only check in the files that have been changed.

Dimensions CM provides both optimistic and pessimistic options. In Dimensions streams, files are writable by default.

NOTE
You cannot enforce either mode, optimistic or pessimistic, of development within Dimensions CM from the server level. Individual Eclipse users may set preferences within Eclipse to facilitate one approach or the other. For example, a user can require a check out before editing files (Pessimistic locking approach).

Ways to Work with Dimensions

You can use any of the following Dimensions features to work with your Eclipse projects. Your organization should choose the appropriate model before adding Eclipse projects to source control and rolling it out to all developers.

- **Dimensions projects**: Use Dimensions projects to explicitly check files out and in when working on them. Projects may be most useful for teams for whom more cautious day to day version management is critical. For example, this may include large teams whose members work concurrently on the same files, or teams working on the main line of development for a product. See the *Dimensions User's Guide* for more information on Dimensions Projects.
- **Dimensions streams**: Use Dimensions streams to manage parallel development efforts that do not require explicit file check out. Streams enable teams working on parallel lines of code to work in a more agile fashion to build and deliver patches and maintenance releases without having to continuously reconcile against the main lines of development. To work with streams, you typically need to:
  - Ensure that all users have added the stream to their local workspace, for details see page 34.
  - As users update local files, they synchronize with the repository to update and merge Repository changes or deliver new or changed local files. For details see page 46.
  - Then, to fetch any changes from the repository to the local workspace, users should regularly update their projects. For details see page 38.
- **Dimensions project containers**: For details see page 84.

Sharing Work Areas with Other Dimensions Clients

When you add Eclipse projects from Dimensions CM to Eclipse you can define a work area outside of your Eclipse workspace location. This area is compatible with all Dimensions clients. You can open the same project in the Desktop or Windows Explorer clients, and update and check in updates all from one common working location. This also applies when sharing a local Eclipse project that is located outside your workspace. In either case, you can define work areas that can be shared by other clients.

Even if you have previously set up Eclipse projects with work areas under the Eclipse workspace, you can use these projects with a shared work area. Dimensions CM provides an auto-sharing feature that allows you to get existing eclipse projects and then import them into Eclipse. For step by step directions see page 87.

Adding Eclipse Projects to Source Control

About Managing Eclipse Projects

To use the version management functionality available in Dimensions for Eclipse you must share your Eclipse projects with Dimensions CM. Sharing projects maps the projects either to single projects or streams in Dimensions, or to subfolders within project containers. When you map a single Eclipse project to a single Dimensions project, the project mappings occur at the root level; the root of the Eclipse project maps to the root of the Dimensions CM project. Dimensions CM attempts to create a project or stream with the same name as the Eclipse project. If the actual Eclipse project name does not meet the requirements for project names in Dimensions CM a modified file name is used. Dimensions CM places the files in a project under Dimensions CM control and a copy of the files are placed into the Dimensions CM repository. You can get or check out the files to your local workspace, where you can edit the files, and then merge or check in your changes into source control.

To learn more about Dimensions projects, streams, and parallel development concepts, see the *Dimensions User's Guide* or *Getting Started Guide*.

TIP You can group multiple shared projects using project groups.

About Repository Projects

In Dimensions CM Explorer, projects are displayed under the **Repository Projects** node. The Repository Projects node lists the project groups and projects that are available on the Dimensions CM server to which you are connected. Projects are further broken down into the following categories:

- **Single Eclipse Projects**— A single Eclipse project is a project or derivation that was added to the repository through the integration and contains Eclipse specific metadata. Derivations include projects or baselines that may not have been directly added to the repository from Eclipse, but were derived from an original project which was added through the integration. See page 27 for more information on single Eclipse projects.
- **Eclipse Project Containers**— An Eclipse project container is a type of Dimensions project or stream that can store multiple Eclipse projects. An Eclipse project that is stored within an Eclipse project container is called a contained project. An Eclipse project container shows the same derivation information as Single Eclipse projects. All repository files from the root of the selected stream or project are added as an Eclipse project.
- **Streams**—Eclipse projects that are associated with Dimensions streams are listed under the Single Eclipse Projects or Eclipse Project Containers nodes with a unique icon.

- **Eclipse Project Groups**—Eclipse project groups are parent level projects that contain multiple subprojects. Project groups are used to organize multiple projects and make it easier for users to manipulate projects.
- **Other Baselines**—Other baselines are Dimensions baselines that do not contain Eclipse specific metadata. You can add these baselines to your local workspace by right-clicking the baseline and selecting Add to Workspace. For more details see page 34.
- **Other Projects**—Other projects are Dimensions projects or streams that do not contain Eclipse specific metadata. You can add these projects or streams to your local workspace by right-clicking and selecting Add to Workspace.

Sharing Projects With Dimensions CM

To place existing Eclipse projects under Dimensions CM control, you must share your Eclipse project with Dimensions CM. If you are working with Eclipse project containers, see page 86.

When you Share an Eclipse project that resides outside of the Eclipse workspace, the location where the files reside becomes a compatible work area that can be shared with other Dimensions clients. That means that Eclipse, the desktop client, and other clients can work on common files located in this shared area. This is not possible for Eclipse projects that reside within the Eclipse workspace.

1 In Eclipse, right-click the project icon and select Team | Share Project.

The Share Project dialog appears.

2 Select **Dimensions** from the **Select a repository type** list and click the **Next** button.

The Select Connection panel appears.

3 Select one of the following choices:

- **Create a new Dimensions connection**, which will allow you to create a new connection to a Dimensions CM server.
- **Use existing Dimensions connection** and choose one of the existing Dimensions CM connections from the list.

4 If your current Eclipse session is not yet logged in to Dimensions, the Dimensions Login dialog box appears. Log in to Dimensions.

5 The Project Selection panel appears.

Select from the following choices. The options available here depending on how your administrator has configured your Dimensions installation. You may be able to use Dimensions project, streams, or both.

- **Create a new Dimensions Stream** to create a new stream in Dimensions and connect it to the Eclipse project.
- **Create a new Dimensions Project** to create a new project in Dimensions and connection it to the Eclipse project
- **Use existing Dimensions Project**, which allows you to choose an existing project from the tree.

- **Use existing Dimensions Stream**, which allows you to choose an existing stream from the tree.

6 To add the project to a Dimensions project container, select **Create in Eclipse Project Container**. For details on working with project containers, see page 84.

7 If you chose to create a new stream or project, you must complete the following steps:

a For a new stream, click **Next**, then select the product, enter a stream name and description, and name the unique branch that will be used to identify file revisions stored in the stream.

b For a new project, click **Next**, then select the product, enter a project name, select a project type, and enter a description.

c To set a container offset, choose the design part, and set a work area for new items, click the **Show Advanced Settings** button. Set the following options:

- **Work Area**: If there is already an established Dimensions work area outside of the Eclipse workspace and above the project location, the project files will be shared into this work area.
  If there is no existing Dimensions work area above the Eclipse project, you can select any folder between the Eclipse project's location and the root of the system drive and create a new compatible work area. **We strongly recommend that you do not use the system root.**
  The container offset is automatically adjusted to ensure that the repository structure matches the path from the work area. For example, if the Eclipse project is located at "C:\Work\MyProjects\Project1"

  And the work area is set to

  C:\Work\MyProjects

  Then the offset will be set to

  Project1
- **Container Offset**: For projects that you are sharing to a new Eclipse project container, you can set the container offset. This offset allows you to modify the project layout in the Dimensions CM repository, if you want to structure it differently from the work area layout. By default this is the project name.
- **Design Part for New Items**: Set the design part for new items. By default, this is set to the top part of the product in which the new project or stream is being created. This option applies to Single Eclipse Projects, and to projects that you are sharing to a project container.

IMPORTANT

- Dimensions for Eclipse performs a number of checks to ensure correct configuration. Error messages appear in the following cases:
  - Pre-existing work areas are not associated with the Dimensions project or stream into which the Eclipse project is being shared
  - Multiple pre-existing work areas reside above the Eclipse project location
  - Pre-existing work areas reside below the project location.

  You can not share in any of these circumstances.

- Although it is possible to specify the root of a drive on Windows or the root folder on Linux as the work area, a warning will appear. It is strongly recommended that the drive root is not chosen as the work area.

  If you are working with project containers, see page 86.

8 Click **Next**.

9 If you are creating a new stream, complete the Additional Details dialog box and click **Next**. If you are creating a new project, skip ahead to the next step. Complete the following options on the Additional Details dialog box:

- From the **Based On** group, choose **Nothing** to create an empty stream, or choose **Based on Stream** or **Based on Baseline** to populate the new stream with items from an existing stream or baseline.
- Under **Change Management Rules**, you can choose whether to require users to specify valid change requests when delivering their local changes to the stream.

10 If you chose to create a new project, complete the following steps:

a Complete the Based-on dialog box and click **Next**:

- Choose the **Nothing** option to create an empty project.
- Choose **Based on another Project** or **Based on Baseline** to populate the new project with items from an existing project or baseline.

b Complete the Options dialog box and click **Next**:

- Under **Version Management Options**, choose whether or not to create branches when new versions of files are created.
- Select **Allow user to override default version number** if you want users to be able to enter non-default version numbers when creating new file versions.
- Under **Change Management Rules**, choose whether to always enable or always disable change management rules in the new project, or to use the change management rules set on item types.
- Select the **Request required to refactor** option if you want to require users to associate a change request to any refactoring activities, such as moving and renaming files.

c On the Named Branches dialog box, select valid named branches for the new project. CTRL+click to select multiple branches. Choose a default named branch from the **Default Branch** list. Click **Next**.

11 If you have selected an existing Eclipse Project Container or Dimensions project or stream, the Share Project page appears. From here you can define a work area, set a container offset, and choose a design part. The same considerations and constraints for creating a work area, offset and design apart apply as described in Step 7, above.

12 If you are working with Eclipse Project Containers, you may be required to specify a request when sharing projects. If the Relate Requests page appears, select a request from your inbox or working list.

13 Review the Sharing Summary dialog box to ensure that the share settings are correct.

Do any of the following:

- Select **Add workspace files to source control** to automatically add the files within the project to Dimensions CM source control.

- Select **Open Synchronize View when done sharing** to display the synchronize view which allows you to synchronize your local files with the Dimensions CM repository. See page 46 for details on using the Synchronize View.

14 Click **Finish**.

Depending on your selections, any of the following may happen:

- A new project or stream is created in Dimensions CM for your Eclipse project, or the Eclipse project is added to a subfolder in a project container.
- The Synchronize View appears allowing you to place files into the Dimensions CM repository.

After your project is placed under control, a gold cylinder is added to the icon of each item to indicate that the item is under source control. The Dimensions CM information for the project, including its product and Dimensions CM project name, appears in brackets alongside the Eclipse project name.

Excluding Files and Directories from Source Control

You can configure the Dimensions for Eclipse integration to exclude specified files from source control. You can configure Eclipse to ignore (exclude) your back up or temp files, which will help eliminate unnecessary files from your Dimensions CM project.

The Dimensions for Eclipse integration has two options for ignoring files from source control:

- **Standard Eclipse Preferences**

  The standard Eclipse preferences ignores files based on a global preference setting. The standard preferences uses a pattern match to determine if files should be ignored. The preferences automatically searches any derived resources, however it has some deficiencies. The standard preferences test only leaf names after containers have been expanded, which means that certain patterns will not get ignored as intended. For example, specifying temp* will ignore files called temp, temp.txt temperance.txt but it will not ignore files in a directory temp or any subdirectories of temp. A second limitation is that specifying temp.txt will ignore it across all projects and directories. It may be necessary to ignore just one particular file with the name.

- **.dmignore File**

  The .dmignore file ignores files and directories based on the entries in the .dmignore file. This file contains project-specific ignore patterns which will be used in addition to the standard preferences. dmignore supports wildcards and pattern matching for both files and directories. A .dmignore file applies to all files and directories in the folder in which it is located; however it doesn't apply to subfolders. For example, if .dmignore contained the entry temp* and you had the following structure:

  ```
  project
  .dmignore
  temp.txt
  temp/file1.txt
  dir1/temp1.txt
  ```

  the temp.txt file and the temp folder (and its subfolder) would be ignored. The temp1.txt file would be included because it is in a subfolder. To ignore this file, you would add an additional .dmignore file in the dir1 subfolder.

To apply ignore rules recursively, add 'r:'. For example, to ignore all `*.xml` files in the current folder and all child folders:

```
r:*.xml
```

To clear all recursive rules inherited from parent `.dmignore` files, add `'c:'`. For example, if a parent ignore file includes:

```
r:*.xml
```

and you add `c:` to the current ignore file, then all `xml` files in the current folder, and all child folders, are *not* ignored.

TIP Place the .dmignore files under source control.

When you perform a source control activity, such as synchronizing a project, the integration runs the ignore checks based on the selection context. The following rules apply to the checks:

- If a resource is under control, it will never be ignored.
- If a resource is linked, it will be ignored.

NOTE If the user selects to add an 'ignored' resource to source control, the resource is added to source control. However, linked resources are hard ignored, meaning they cannot be added to source control.

Excluding Files Using the .dmignore File

You can use the .dmignore file to exclude files from source control activities. This file can be used independently or in addition to the standard Eclipse ignore preferences. See page 32 for more information.

NOTE You can still explicitly add files to source control even though you have added them to the ignore lists.

To exclude specified files or folders using the .dmignore file, either:

- Right-click on the file or file type and choose **Team | Add to .dmignore**. Choose if you want to exclude only this file name, all files with this extension, or a custom pattern (you can use wild cards such as * and ?). You can also apply the rule recursively.

  Click **OK** and the entry is added to the .dmignore file in the root of your project. If the .dmignore file does not exist, it will be created.
- Open the .dmignore file and add the values and patterns which you do not want to be added to source control. You can use wild cards such as * and ?.

NOTE

- The .dmignore file applies to all files and directories at the same level as the file.
- The .dmignore file can contain an entry referencing itself, .dmignore, if you want to prevent the file from being added to control.

Excluding Files Using Standard Eclipse Preferences

You can use the standard Eclipse preferences to exclude files from source control. The preferences can be used independently or in addition to the .dmignore file. See page 32 for more information.

NOTE You can still explicitly add files to source control even though you have added them to the ignore lists in Dimensions projects.

To exclude specified files using the standard Eclipse preferences:

1. Select Window | Preferences. The Preferences dialog box appears.
2. Click the plus sign next to **Team** and select **Ignore Resources**.
3. Examine the **Ignore Patterns** list for the file type you wish to exclude from source control. If it is not listed, do the following:
   a. Click the **Add** button. The Enter Ignore Pattern dialog box appears.
   b. Enter a pattern that defines a file type to ignore, you can use these wildcards:
      - Asterisk (*): represents one or more characters.
      - Question mark (?): represents one character.
   c. Click **OK**.
4. In the **Ignore Patterns** list, ensure that a check mark appears next to each file type you want to exclude from source control.
5. Click **OK**.

Adding Repository Projects and Streams to your Workspace

Adding Projects to your Workspace

To work on a project or stream that already exists in the Dimensions CM repository do one of the following:

- Add the project to your local Eclipse workspace.
- (If you are adding single or multiple projects from an Eclipse project container) Add the projects to a location outside of your Eclipse workspace that can be shared with other Dimensions clients, such as the desktop client.

To add existing repository projects to your Eclipse workspace or a shared work area:

1. In Dimensions CM Explorer expand the Repository Projects node.
2. Select any of the following:
   - A single project in **Single Eclipse Projects**. The Eclipse project name will be taken from the `.project` file or from the metadata.

- One or more projects in **Eclipse Project Containers**.
- An existing project under **Eclipse Project Groups**. This adds all of the child projects to your workspace. The Eclipse project name is taken from the `.project` file or from the metadata.
- A project or baseline under **Other Baselines** or **Other Projects**. If there is not a project file a name is generated from `product_projectid` for the new Eclipse project. You can change the automatically generated name.

3 Right-click and select **Add to Workspace**. The wizard walks you through the process of adding existing streams and projects to your Eclipse workspace.

4 If you are adding an Eclipse project container do one of the following:

- To add all of the child projects to your workspace select **Nested Eclipse projects** and click **Next**. Select the projects that you want to add to your workspace, click **Next**, and go to step 6.
- To choose how the project will be added to Eclipse select **Project container**, click **Next**, and go the next step.

5 Select one of these options:

- **Add and configure the project using the Eclipse New Project Wizard**

  Enables you to specify a project type and name, the default location, working sets, and referenced projects. Click **Finish** and use the New Project Wizard to add the project.

- **Add as project**

  Enables you to add the entire project or stream as an Eclipse project. Not available for shared Eclipse projects or streams that have a `.project` file generated by Eclipse. Select a project from the list and click **Next**.

6 Select the project location. If you do not want to use the default, unselect **Use default workspace location** and enter the path to the shared location. This may be an existing work area that is compatible with Eclipse and other projects. If you choose a new location a new compatible work area is created.

NOTES

- If you are adding a single project check that the path concludes with the correct offset of the project in the Dimensions project or stream.
- If you are adding multiple projects the location is the root location and all projects will be located at their offsets relative to this location.
- For both cases a compatible work area is created. Local paths relative to the work area location match the offsets in the Dimensions project or stream. If there is a pre-existing compatible work area above this location, the location will be automatically adjusted to match.

7 Click **Finish**.

IMPORTANT!

Dimensions for Eclipse checks to ensure the correct configuration. Error messages appear in the following cases:

- Work areas already exist that are not associated with the Dimensions project or stream from which the Eclipse project is being added

- Multiple pre-existing work areas reside above the project location

You cannot add to the workspace if either of these conditions occurs.

It is possible to set the location to the root of a drive on Windows or the root folder on Linux. A warning appears and it is strongly recommended that you do not choose a drive root.

Changing the Source Control Connection

Complete these steps when you want to change the connection from one source control project to another. Changing the connection changes the Dimensions project that the Eclipse project is mapped to. It allows you to work with other contributors' projects or streams. It does not change the workspace files to match the contents of the new mapping. This must be carried by synchronizing with the repository.

NOTE Not available for streams.

To change the source control connection:

1 Select **Window** | **Show View** | **Navigator**.

2 In the **Navigator** view right-click the project and select **Properties**.

3 In the Properties dialog box select **Dimensions**.

4 Click **Switch to Project / Baseline**. The Switch dialog box appears.

5 Do one of the following:

- Select **Find** and then **Project** or **Baseline**. Click **Find** to search for a specific project or baseline. If you choose a project container you can select a contained project from the **Contained project** list.
- Select **Browse** and locate the project or baseline.

6 Click **Finish**.

TIP You use also use the context menu on an Eclipse project node.

NOTE This option allows multiple Eclipse projects to be switched in one operation provided they share an Eclipse project container on the same connection. Only one single Eclipse project can be switched at one time. If multiple Eclipse projects are chosen they will switch to the corresponding Eclipse project in the new project. If there is no corresponding Eclipse project they will not be switched.

If you are using the Wind River Workbench product the hierarchy of the Wind River projects is used to determine the set of Eclipse projects to switch.

Disconnecting Projects from Source Control

When you disconnect a project from source control, Dimensions does not delete revisions in the repository; it simply removes the association between the project and the repository. You can choose whether to keep or remove the Dimensions CM metadata about the project.

To disconnect a project from source control:

1 Right-click on a project node, and select Team | Disconnect.

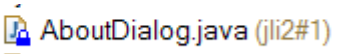
A confirmation dialog appears.

2 To delete the metadata in addition to disconnecting the project, select "Also delete Dimensions CM metadata from the file system." This will delete the information in the .metadata directories in the workspace project, which Dimensions CM uses to store data about files fetched or checked out of Dimensions CM. These directories are created when you get files from Dimensions to store information about the workspace files and directories. They are updated as source control operations are performed.

3 Click **Yes** to confirm the action.

File Status Glyphs

In the Eclipse Explorer views, file icons are modified to represent their current source control status. Refer to the table below for examples.

Glyph	Status
	Under source control. A gold repository glyph appears on the file / folder icon, for example: 
	Checked out to you. A check-mark appears on the file / folder icon, for example: 
	Checked out to multiple users.
	Multiple revisions are checked out to different users, but not to you.
	Checked out to another user.
	Modified locally. If the file is checked out and has been locally modified, a pencil and checkmark appear on the file / folder icon, for example:: 
	Exclusively checked out. This appears if the item is set to allow just one revision to be checked out at a time.
	Locally deleted.
	Locked by another user in the stream.
	Locked by the current user. For example: 

Glyph	Status
	Deleted in the repository by another user.
	Locally writable.
c (2) [3]	Newer revision exists in the repository. Bracketed revision number appears to the right of the current local revision number.

Checking Out Projects

When you check out a project, writable copies of the latest revision of files in the project are placed in the work file location and assigned the next revision number.

To check out a project:

1 Right-click the project and select Team | Checkout.

The Check Out dialog box appears with a list of selected files.

2 Modify the panels as necessary.

3 Click **Finish**. A check mark ( ) appears next to each file icon to indicate that the files are checked out.

Checking In Projects

When you check in a project, a read-only copy of the work files are left in the location and the revisions are checked into the Dimensions CM repository

To check in a project:

1 Right-click the project and select Team | Checkin.

The Check In dialog box appears with a list of selected files.

2 Modify the panels as necessary.

3 Click **Finish**.

On successful completion, the check mark is removed from each file icon to indicate that the files are checked in.

Updating a Workspace

When you update a project or stream your local files are refreshed to match the files that have changed in the repository. If you modified a file locally and newer revisions exist in the repository, an error message will appear notifying you of the conflicts and suggesting that you use the synchronize view to merge the differences.

You can update your Eclipse workspace without delivering changes to the repository.

To update a project:

In **Package Explorer** right-click the project and select **Team** | **Update**.

To update a stream:

1. In **Package Explorer** right-click the stream and select **Team** | **Update**. The Update Stream wizard appears. The **Update changes from this stream** box displays the stream ID that will be the source for this update.
2. All the Eclipse IDE projects currently in your workspace are listed. These are the projects that will be updated. To change the scope of the update do the following:
   a. Click **Select**.
   b. Select a projects group from your workspace.
   c. Click Next.
   d. Select the workspace projects that you want to update.
   e. Click **Finish**.
3. To interactively (manually) verify the update results before applying them to the work area, select **Perform an interactive update**.
4. (Optional) Click **Advanced**.
5. Select these options where applicable:
   - Apply the repository date and time to the files.
   - (Typically used when shelving a stream) To reset all changes in the work area and synchronize with the repository select **Reset the work area to the latest repository content**. You can also delete locally added files, such as artifacts added by a local build process.
   - Automatically merge files whose content does not conflict.
   - Show a summary of the results of the operation (only available for non-interactive updates).
   - Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

   Click **Next**.
6. Select a version of the source stream.
7. To run the update click **Finish**. The results of the update are displayed. Use the Dimensions Stream Update view to compare revisions, resolve conflicts, and synchronize with the repository.

Updating a Workspace from Requests

You can update your workspace with requests that are related to the home stream.

TIPS To update from other streams or projects use Merge, see page 53.

1. To open the Requests view do one of the following:
   - Right-click the connection node in Dimensions CM Explorer and select **Show Requests**.
   - From the Dimensions menu select **Show Requests**.
2. To select requests do one of the following:

- Streams: select one or more requests, right-click, and select **Update Stream from Request**.
- Projects: select a request, right-click, and select **Update Project from Request**.

The Select Update Options wizard opens.

3 The **Update changes from these request(s)** box displays the requests that you selected. To change or add requests click **Select** and use the Request Select wizard.

4 If the requests are related to the same stream, the **Include changes from this stream** box automatically displays the stream ID.

If the requests are related to different streams, click **Select**, and select a stream. The Update operation can only update from one stream at a time. If your workspace contains multiple Eclipse projects associated with different streams, run Update again for each stream.

If you selected request(s) that do not belong to the stream associated with your workspace you are prompted to Merge.

5 **The Update changes into these Eclipse project(s)** table lists all the Eclipse IDE projects currently in your workspace. These are the projects that will be updated. To change the scope of the update do the following:

a Click **Select**.

b Select a projects group from your workspace.

c Click Next.

d Select the workspace projects that you want to update.

e Click **Finish**.

6 To interactively (manually) verify the update results before applying them to the work area, select **Perform an interactive update**.

7 To include all items that are related to child requests of the selected change request, select **Also include items related to child request(s)**.

8 (Optional) Click **Advanced**.

9 Select these options where applicable:

- Apply the repository date and time to the files.
- Automatically merge files whose content does not conflict.
- Show a summary of the results of the operation (only available for non-interactive updates).
- Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

10 To run the update click **Finish**. The changes that are going to be added to the workspace are displayed. Accept or change the suggested resolutions.

11 Click **Finish** to update the workspace with the changes associated with the selected requests.

Creating New Streams

You can create the following types of streams:

- An empty stream that initially contains no items.
- Based on a baseline.
- Based on any version of another stream. The new stream is a child of the parent stream from which it was created.
- Based on the items that are currently under work in your workspace. The new stream includes the item revisions from the current stream in Dimensions, not any local changes that have not yet been checked in or delivered to Dimensions. For example, if you updated your workspace with the most recent revision (1.5) of `readme.txt` and have made some local edits to this file but have not yet delivered those edits to a new revision in Dimensions (1.6), the new branch stream will include revisions up to 1.5.
- A personal stream, a private development branch that is only visible to the originator.

Personal Streams

Personal streams are private development branches that are only visible to the originator. Personal streams enable you to:

- Isolate work from an existing public stream.
- Work in dedicated personal stream until your changes are ready to be merged back into a public stream.
- Create a new stream based on a work area. You are working on changes in a work area but the changes cannot be delivered to the associated stream. For example, the stream may be locked or the feature has been dropped from the release. To shelve your work you can create a new personal stream based on the work area, see page 57.

Personal streams are different to regular streams:

- By default any user can create a personal stream.
- Only the stream originator can view, and work with, a personal stream. However a user with the appropriate permissions (by default the ADMIN group) can delete a personal stream.
- By default CM Rules are disabled for personal streams therefore users do not have to specify requests when delivering changes to personal streams. This behavior can be changed by an administrator.

Like regular streams, personal streams are versioned so if you deliver multiple times you can restore to any previous state. Personal streams have a unique icon to differentiate them from regular streams and projects.

To create a personal stream, create a new stream and select the appropriate option. After work on a personal stream is complete you can merge it into a regular stream, see page 53.

Deleting a personal stream is the same as for regular steams. To avoid taking up space in your database, and to optimize performance, the recommended best practice is to delete all personal streams that are no longer required.

To restore a personal stream to a clean work area, in Dimensions CM Explorer right click the personal stream and select Add to Workspace.

Creating a New Stream

1 Do one of the following:

Dimensions CM Explorer

- To create an empty stream right-click the top level connection node and select **New** | **Stream**.
- To base the new stream on the contents of an existing stream, right-click it and select **New** | **Stream**.

Package Explorer

To create a new stream based on the items that are currently under work in your workspace, right-click the stream and select **Team** | **Branch from Workspace**. The Project Branch screen of the Create Stream wizard appears. Do the following:

- To create a baseline of the initial state of the new stream select **Baseline new project**.
- To start working on the new stream when it is created select **Start working on the new project**.
- Click **Next**.

2 On the General page:

- Select a product.
- Enter a name and description for the new stream.
- In the **Unique Branch Name** box enter the branch name to be used for new items created in this stream.
- To create a personal stream select **Create a personal stream**. Personal streams are private development branches that are only visible to the originator. For details see page 41.
- To add the new stream to your list of Favorites select **Add the new stream to my favorites**. Favorites are shared across all clients.

3 Click **Next**.

4 Select one of the following:

- **Nothing** if you want the stream initially to contain no items.
- **Based on Stream** if you want the new stream to be populated with the item revisions from an existing stream. Click **Find** and use the Select Stream wizard to search for one.

  Optionally select a version of this stream on which to base the new stream.
- **Based on Baseline** if you want the new stream to be populated with the item revisions from a baseline. Click **Find** and use the Select Baseline wizard to search for one.

5 Select **Valid request must be specified when delivering changes** if you want CM to require a request to be entered when any changes are made to the stream.

6 Click **Next** to review a summary of the new stream's properties.

7 Click **Finish** to create the stream.

Delegating a Personal Stream

You can delegate a personal stream to another user and change ownership of the stream. For example, you are switching to a different task and want to delegate the changes in a personal stream to another user.

NOTE

- After you delegate a personal stream that you originated you can continue contributing to it. Access is controlled by the privilege *Control Personal Stream.*
- All users in the ADMIN group have this privilege granted by default.

1 In Dimensions CM Explorer select one or more personal streams, right-click, and select **Delegate**.

2 In the Delegate Personal Stream dialog box select a user.

3 Click OK. The personal streams are now delegated to the user you specified.

Creating New Projects

You can create the following types of projects:

- An empty project that initially contains no items.
- Based on a baseline.
- Based on any version of another project. The new project is a child of the parent project from which it was created.
- Based on the items that are currently under work in your workspace. The new project includes the item revisions from the current project in Dimensions, not any local changes that have not yet been checked in or delivered to Dimensions. For example, if you updated your workspace with the most recent revision (1.5) of `readme.txt` and have made some local edits to this file but have not yet delivered those edits to a new revision in Dimensions (1.6), the new branch project will include revisions up to 1.5.

Creating a New Project

1. Do one of the following:

   Dimensions CM Explorer

   - To create an empty project right-click the top level node and select **New** | **Project**.
   - To base the new project on the contents of an existing project, right-click it and select **New** | **Project**.

   Package Explorer

   To create a new project based on the items that are currently under work in your workspace, right-click the stream and select **Team** | **Branch from Workspace**. The Project Branch screen of the Create Project wizard appears. Do the following:

   - To create a baseline of the initial state of the new project select **Baseline new project**.
   - To start working on the new project when it is created select **Start working on the new project**.
   - Click **Next**.

2. On the General page:
   - Select a product.
   - Enter a name for the new project.
   - Select a workset type.
   - Enter a description of the new project.
3. Click **Next**.
4. Select one of the following:
   - **Nothing** if you want the project initially to contain no items.
   - **Based on another project** if you want the new project to be populated with the item revisions from an existing project or stream. Click **Find** and use the Select Project wizard to search for one.

     Optionally select a version of this stream or project on which to base the new project.
   - **Based on Baseline** if you want the new stream to be populated with the item revisions from a baseline. Click **Find** and use the Select Baseline wizard to search for one.
5. Click **Next**.
6. Select version management options and change management rules.
7. Click **Next**.
8. In the **Valid branches** field select one or more branches for your project.
9. From the **Default branch** list select a branch.
10. Click **Next** to review a summary of the new project's properties.

11 Click **Finish** to create the project.

Comparing and Synchronizing Projects

Comparing the Current Project With the Repository

You can compare the current project in your workspace with a specific baseline and project in the Dimensions CM repository. The comparison displays which files have changed. You can compare from multiple views, including the Project Explorer, and Java and Package Explorer view.

To compare a project with a baseline from the Project Explorer view:

1 Right-click the project and do one of the following:

- Select **Compare With** | **Latest from Project** to compare the local files with the latest files in the repository.
- Select **Compare With** | **Another Project or Baseline** to compare with another project or baseline. In the compare dialog do one of the following:
  - Click **Find** and specify criteria to search for an existing baseline or project in Dimensions CM.
  - Click **Browse** to look through the tree of projects and baselines and select a project or baseline to compare with.

2 Click **Finish**. You can compare the differences and commit them to the repository.

NOTE If you select Latest from Project for a stream, the Dimensions Stream Compare view opens. You can use this view to commit changes but to merge you need to perform an update.

To compare from the Dimensions CM Explorer view:

1 Select either two projects, two baselines, or a project and a baseline from the explore tree.

2 Right-click and select **Compare**.

The Dimensions for Eclipse integration will attempt to resolve the common ancestor for the two items. If a common ancestor cannot be found, you will be prompted to choose a common ancestor or continue without one.

About Synchronization

Synchronizing allows you to verify that the files in your local workspace match the files in the Dimensions CM repository. In a multi-user environment, synchronizing your workspace assists you with:

- Removing files from your local workspace that other users have removed from the source control project.
- Adding files to your local workspace that other users have added to the source control project.

- Adding files to the source control project that you have added to your local workspace.
- Updating the content of files in your local workspace that other users have modified and checked into the source control project.
- Committing the changes that you have made in your local workspace to the source control project.

Synchronization marks which files have been changed, inserted, moved, renamed, or deleted. You can resolve the discrepancies by importing changes to your local workspace, merging the different files, or committing the changed files into source control. Your changed files are either checked in or optimistically updated, which means that they are checked in without being checked out first.

If the repository revision of the file to be checked in has been changed by someone else, you will be required to merge the files to your local workspace, and then check in the merged file.

Dimensions for Eclipse uses metadata to determine if folders or files have been moved or renamed. You can commit these changes to source control from the synchronize view.

Synchronizing Your Workspace with a Dimensions CM Repository

The Synchronize with Repository command compares the local workspace to the repository and lists all incoming and outgoing changes, including updates, deletions, and conflicting changes. Conflicting changes are when both the local file and the repository file have been changed.

You can resolve the discrepancies by importing changes to your local workspace, merging the different files, or committing the changed files into source control.

NOTE Full synchronization is only supported for projects.

To synchronize your workspace with source control:

1 Select the projects that you want to synchronize with the Dimensions CM repository.

2 Right-click and select Team | Synchronize with Repository.

The Team Synchronizing perspective appears.

3 Use the features in the Team Synchronizing view to evaluate the files that differ from source control, and choose whether to synchronize, check in, or merge the changed files. This view displays file status that you can use to decide how to act on specific files. For example, this view identifies foreign items, which are files that are related to items in a different project or stream.

NOTE If you are using the Wind River Workbench product, the Hierarchy of the Wind River projects will be used to determine the set of Eclipse projects to synchronize.

About the Team Synchronize View in Eclipse

The Team Synchronize view in Eclipse displays when you select **Synchronize with Repository**. The Synchronize view allows you to inspect the differences between the local Workbench resources and their remote counterparts, as well as update resources in the

local Workbench or commit changes/resources from the local Workbench to the Dimensions CM repository.

NOTE Full synchronization is only supported for projects.

The Team Synchronize view has many features that help you to perform the synchronization with the repository, such as:

- **Commit Moves**—The commit move operation updates the Dimensions CM repository with information from files which have been moved or renamed.
- **Synchronization states**—Synchronization states show the state of resources in your local workspace compared to those residing the Dimensions CM repository.
- **Modes**—The different synchronization modes (Incoming, Outgoing, Incoming/ Outgoing, Conflicts) allow you to filter the differences, helping you to find the differences to be resolved based on your company's work flow.
- **Conflict Handling**—Conflicting resources can be merged (instead of overwriting) using Team Synchronization by performing either an update or a compare to merge the differences. You must use a Compare editor to compare and resolve conflicts. A Compare editor can be opened by choosing **Open in Compare Editor** from the context menu. The Compare editor allows you to manually resolve the conflicts in the file. Once completed, perform a **Mark as Merged** on the conflict to indicate that you are done. This will change the conflict into an outgoing change.

Viewing Project Properties

Dimensions CM Explorer allows you to access project attributes, history, relationships, and default working branch through a single dialog.

Procedure **To view source control information:**

In Dimensions CM Explorer, right-click the project node and select **Properties**.

The Properties dialog appears for the project.

On this dialog, you can choose the Project Working Request or the Project Working Branch for the project:

- The **Project Working Request** is the default request that is used to be related *in response to* new item revisions that you create in this project.
- The **Project Working Branch** is the default branch on which new item revisions that you create in this project will be placed.

From the project properties page, you can also select the default library cache area. Selecting a library cache area for your project can increase the response time when contacting Dimensions CM. See the *Dimensions CM User's Guide* for more information on library cache areas.

Viewing the History of a Project

The history of a project shows the previous actions on the project, such as the addition of files or the creation of baselines. The Project History is the history of a specific Dimensions CM project. It contains all file changes, similar to the Dimensions CM Desktop client. The history of an Eclipse Project is all the projects and baselines logically in the Eclipse project, such as derivations.

To display the history of a project, in Dimensions CM Explorer, right-click the project node and choose **CM Project History**. The list of actions appear in the **CM Project History** tab.

Merging Projects

IMPORTANT! Merging projects in Eclipse uses the Eclipse merge tool. The Eclipse merge tool does not support refactoring and is therefore not recommended for use with Dimensions projects. For best results, merge projects using the Project Merge tool in the Dimensions desktop client. Or, use Dimensions streams instead of projects.

The Project Merge option allows a selected local project to be merged with a project from the repository. The projects must be related by being derived from the same source, such as when the branch of a project is merged into the mainline of development.

The two projects are merged into your local workspace. The merge is performed using the compare and merge capability of Eclipse.

For the pedigree to contain merge transition, **Mark as Merged** has to be selected on a file or files which have already been merged and saved locally. When **Mark as Merged** is used, the merged files are removed from the merge view.

The merged files are committed to the Dimensions CM repository using the **Synchronize with Repository** command. The merge is recorded in Dimensions CM only when merge changes are committed to the repository.

Note that there is no lock held on the resources being merged, so other users can make changes to the repository.

To merge projects, right-click on the project to merge and select **Team | Merge**.

Working Offline

You may not be able to immediately connect to the Dimensions CM database due to lack of network access. In these situations, you can edit your files in your local workspace by changing them to local mode and then checking in the files when you have access to the Dimensions CM repository.

A second method of placing files into local mode is to choose the **Work Offline** option when you are prompted to connect to the Dimensions CM server.

This enables the Work offline option for your editing session. Working offline automatically changes files to local mode when you attempt to edit them. The work offline session ends either:

- when you explicitly connect to an offline connection. This allows you to check in your changes without exiting the IDE.
- when you exit your Eclipse workbench. The next time that you start your Eclipse workbench, you will be able to log in to Dimensions CM and check in your changes.

Delivering Changes from a Workspace to a Stream

If your team uses streams, use the deliver operation to commit changes from your workspace to a Dimensions CM repository.

TIP The deliver operation only supports commits. If there are conflicting changes use **Team** | **Update** or update directly from the view.

To deliver changes to a stream:

1. In **Package Explorer** right-click a stream and select **Team** | **Deliver**. The Commit wizard appears.
2. On the Commit Changes to Repository page select the resources that you want to deliver. Click **Next**.
3. On the Relate Requests page select change requests from the My Inbox and My Working List tabs. Click **Next**.
4. On the Item Attributes page select attributes for the revisions your are delivering.
5. Click **Finish**.

Undoing Changes in a Stream or Project

You can roll back the delivery of files and folders to a stream or project. For example, you deliver changes but discover problems in the code and decide to remove the changes. Undo creates a new changeset with the reverted changes that preserves the full history.

NOTE

- Requires the same privileges as deliver.
- Undoes an entire changeset, not individual item revisions.
- The original request relationships are not undone.
- Only makes changes in a repository and does not affect the items in a work area.
- After completing an undo, to synchronize your work area with the repository, run an update.
- You can run undo at any location as long as the changeset does not affect items further up the hierarchy.
- Can only undo a single changeset.
- If items in a changeset have more recent changes in a newer changeset, you cannot undo it.

Assume that changeset 14 has this item revision:

```
f.txt revision #2
```

And that changeset 13 has the previous version of the same item:

```
f.txt revision #1
```

If you undo changeset 14, the revision in changeset 13 becomes the tip revision and a new changeset, 15, is created with the change:

```
f.txt revision #1
```

1. Do one of the following:
   - To show changes for an entire stream or project:
     - In Dimensions CM Explorer right-click a stream or project and select Changesets.
     - In Package Explorer right click the root of a Java project and select Team | Changesets.
   - To only show changes for a subfolder, in Package Explorer right-click the subfolder and select Team | Changesets.
2. In the Changesets dialog, right-click a changeset and select Undo Stream/Project Version.
3. (Optional) Select requests to relate to the change and add a comment.
4. Click OK.

Opening Micro Focus PulseUno

Micro Focus PulseUno is a web-based tool that enables development teams to examine the health and quality of their changes. You can open the PulseUno home page for the current user in an embedded browser in a new view.

Do one of the following:

- From the Dimensions menu select PulseUno.
- In Dimensions CM Explorer, right click a Dimensions connection and select PulseUno.

Displaying Reviews

Reviews are part of PulseUno that enable you to:

- Comment on, and review, the code changes in your development projects.
- Collaborate with team members.
- Get insight into the health of changes in your changesets and streams, such as the results of chain runs.
- Promote team work and development best practices.
- Vote to approve or reject reviews, which may cause reviews to be marked as approved or sent for rework.

Reviews are integrated into your client or IDE and enable you to:

- Display reviews in a separate view. You can manage a review, such as adding comments and changing its state.
- Open the review related to a specific request or changeset.

See also: PulseUno help

- To display the review associated with a changeset, in the Changesets view, click the health icon.
- To display the review associated with a request, in the Requests view, right click a request, and select Open Review.

Browsing the Repository

You can browse a the contents of a CM repository inside your IDE and view the files and folders in your streams and projects. In the repository browser you can:

- View the tip version or previous versions.
- View the code in each file.
- Add review comments to a files. Comments are attached to that version of the file and are displayed on any associated Review pages.

1 Do one of the following:

- In Dimensions CM Explorer, right click a stream or project and select Open Repository Browser.
- In Package Explorer, right click a Java project and select Team | Open Repository Browser.

The browser opens in a new view and displays the contents of the latest changeset.

2 Navigate to a folder or file to view its contents.

3 To view the contents of a file without the color coded syntax, click Raw.

4 To browse the entire contents of a previous changeset, select the History tab, select a changeset, and click View at Version.

5 On the Content tab, browse the files and folders in the changeset.

6 To go to the tip of the stream or project, click View at Tip.

The repository browser is an embedded version of PulseUno's Code view.

See also: PulseUno help

Topic Streams and Pull Requests

Topic streams are temporary streams that you use for a set of defined changes, for example, to fix a defect or develop a small feature. Topic streams enable you to isolate your changes from a mainline. If you need to suspend work that is in progress, or isolate work into a dedicated stream, you can shelve the changes from a work area into a topic stream.

A pull request is a special type of review that you use to evaluate and merge a set of defined changes, typically in a topic stream.

A topic stream and its related pull request work together to help you manage and merge changes:

- A topic stream is created from a parent stream, which creates a pull request.
- Changes are delivered to the topic stream and related to the pull request.
- The pull request is reviewed, approved, and merged into the topic stream's parent.

When you create a pull request, you assign reviewers to it. Reviewers see the code changes on the Changes tab of a pull request. They can add comments about the changes, identify issues, make suggestions, and answer questions. When a comment is made, the pull request author and reviewers receive an email with a snippet of the code and the review comments.

When changes have been reviewed and approved, and if there are no conflicts between the topic stream and its parent, you can merge the pull request. This merges the contents of the topic stream into its parent stream without using a work area.

Rehome is a quick and easy way to switch a work area from one stream to another.

If you have a topic and parent stream that both contain changes, and you want to update the topic stream with the changes from its parent, you can rebase the topic stream.

For detailed information about topic streams and pull requests, including examples and scenarios, see the Dimensions CM client user help.

Merging Changes across Streams

You can merge:

- Changes across streams or their folders.
- Changes from a baseline into a target stream.
- Changes from requests owned by another stream into a target stream.

Merge is similar to Update and uses a work area to apply changes and process conflicts. This enables you to safely merge, build, and test before delivering the merge results to a target stream in a repository.

When you merge two objects, for example a stream with another stream, or a stream with a baseline, Dimensions CM looks for a common ancestor. This is a point in time where the streams or work areas diverged. This is typically referred to as a three-way merge. For example, assume you have two streams that were created from the same baseline. Modifications are performed in both streams by different developers and delivered to CM. You now merge stream 1 into stream 2. Dimensions CM looks at the baseline (the common ancestor), finds the modifications that were delivered to stream 1 after that point, and merges those changes in stream 2.

The merge process uses changesets to find the common ancestor. A changeset is a logical grouping of changes that is created automatically every time that you deliver changes to a stream or project in a repository. For details see page 59.

For merge examples and use cases see the chapter *Merging Changes across Streams* in the *Dimensions CM User's Guide*.

Merging Specific Changes between Streams

When you merge changes between streams you can select changes that are related to specific requests, commonly known as *cherrypicking*. For example, you have delivered multiple sets of changes to a stream but want to merge one specific set into a different stream. For more details and examples see the *Dimensions CM User's Guide*.

Merging Changes across Streams

1 Do one of the following:

- In **Package Explorer** right-click a stream or folder and select **Team** | **Merge**. The Merge wizard opens. Select **Merge streams or their folders** and click **Next**. To select the source stream do one of the following:
  - Select the **Find** option, click **Find**, and use the Select Stream wizard to specify the stream.
  - Select the **Browse** option and select a stream from the Repository Projects tree.

  Click **Next**.
- In **Dimensions CM Explorer** right-click a source stream and select **Merge**. The Merge wizard opens.

The **Merge changes from this stream** box displays the stream ID that will be the source for this merge.

2 A list all the Eclipse IDE projects currently in your workspace is displayed. These are the project(s) that you will merge into. To change the scope of the merge do the following:

a Click **Select**.

b Select a projects group from your workspace.

c Click Next.

d Select the workspace projects that you want to merge into.

e Click **Finish**.

3 To interactively (manually) verify the merge results before applying them to the work area, select **Perform an interactive merge**.

4 (Optional) Click **Advanced**.

5 Select these options where applicable:

- Apply the repository date and time to the files in the merge.
- Automatically merge files whose content does not conflict.
- Show a summary of the results of the operation (only available for non-interactive merges).
- Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

Click **Next**.

6 Select a version of the source stream.

7 To run the merge click **Finish**. The results of the merge are displayed in the Synchronize view. Use the view to compare revisions, resolve conflicts, and synchronize with the repository.

Merging Changes from a Baseline into a Stream

1 Do one of the following:

- In **Package Explorer** right-click a stream or folder and select **Team** | **Merge**. The Merge wizard opens. Select **Merge a baseline into a stream** and click **Next**.

  To select the baseline that will be the source for this merge, do one of the following:

  - Select the **Find** option, click Find, and use the Select Baseline wizard to specify a baseline.
  - Select the **Browse** option and select a baseline from the Repository Projects tree.

  Click **Next**.

- In **Dimensions CM Explorer** right-click a baseline and select **Merge**. The Merge wizard opens.

The **Merge changes from this baseline** box displays the baseline that will be the source for this merge.

2 A list all the Eclipse IDE projects currently in your workspace is displayed. These are the projects that you will merge into. To change the scope of the merge do the following:

a Click **Select**.

b Select a projects group from your workspace.

c Click Next.

d Select the workspace projects that you want to merge into.

e Click **Finish**.

3 To interactively (manually) verify the merge results before applying them to the work area, select **Perform an interactive merge**.

4 (Optional) Click **Advanced**.

5 Select these options where applicable:

- Apply the repository date and time to the files in the merge.
- Automatically merge files whose content does not conflict.
- Show a summary of the results of the operation (only available for non-interactive merges).
- Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

6 To run the merge click **Finish**. The results of the merge are displayed in the Synchronize view. Use the view to compare revisions, resolve conflicts, and synchronize with the repository.

Merging Changes from Requests Owned by Another Stream

1 Do one of the following:

- In Package Explorer right-click a stream or a subfolder and select **Team** | **Merge**. The Merge wizard opens. Select **Merge change requests from another stream** and click **Next**. In the **Merge changes from these request(s)** box specify the request changes to be the included in this merge. Click **Select** and use the Request Select wizard to find one or more requests.
- Open the Requests view: right-click the connection node in Dimensions CM Explorer and select **Show Requests** or from the Dimensions menu select **Show Requests**. Select one or more change requests and select **Merge Request into Stream**. The Merge Request wizard opens.

2 If the requests are related to the same stream, the **Include changes from this stream** box automatically displays the stream ID. If the requests are related to different streams, click **Select**, and select a stream. The Merge operation can only merge from one stream at a time. Run 'Merge from request' separately for each steam to which requests are related.

3 All list of all the Eclipse IDE projects currently in your workspace is displayed. These are the projects that you will merge into. To change the scope of the merge do the following:

a Click **Select**.

b Select a projects group from your workspace.

c Click Next.

d Select the workspace projects that you want to merge into.

e Click **Finish**.

4 To interactively (manually) verify the merge results before applying them to the work area, select **Perform an interactive merge**.

5 (Merge change requests only) To include all items in the merge that are related to child requests of the selected change request, select **Also include items related to child request(s)**.

6 (Optional) Click **Advanced**.

7 Select these options:

- Apply the repository date and time to the files in the merge.
- By default only the item revisions associated with the request(s) that you specified will be merged. To merge *all* item revisions that are related to the selected request, unselect **Cherrypick file changes**. For details see page 53.
- Automatically merge files whose content does not conflict.
- Show a summary of the results of the operation (only available for non-interactive merges).
- Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

8 To run the merge click **Finish**. The results of the merge are displayed in the Synchronize view. Use the view to compare revisions, resolve conflicts, and synchronize with the repository.

Shelving Streams

About Shelving

Shelving enables you to store local changes in a personal stream in a repository and optionally remove the changes from a work area. You can use shelving to:

- Shelve changes and reset a work area

  You can remove changes from a work area that you do not want to deliver to the associated stream when:

  - The changes are not stable or complete.
  - You need to interrupt your current work and set aside the changes.
  - You need to switch to a different task but want to use the same work area rather than populate a new one, which can be time consuming.

  After the shelving operation is complete you reset the work area to the latest repository content.

- Backup changes

  When you are sharing a stream with other developers but are not ready to deliver your local changes, you can backup your work by committing it to a personal stream. This creates a snapshot of the work area at that point in time. When creating backups you keep the local changes and do not need to reset the work area.

When you are ready to resume work on the shelved changes you can restore them by merging the personal stream back into the work area.

Typical shelving scenario:

- Your team lead asks you to improve functionality in the next release of an application.
- You download the tip of the stream to a local work area and begin work. Over the course of the next few days you create new files and modify existing ones.
- Your manager asks you to interrupt your work and fix a very high priority bug that affects the same functional area. The fix requires you to modify some of the files that you recently changed. To avoid conflicts between the recent changes and the bug fix, and to use the same work area, you shelve your work to a personal stream and reset your work area to the latest repository content. You now have a clean work area that does not contain the changes you previously made.
- You fix the bug and deliver the changes to the stream.
- Your manager asks you to resume work on improving the functionality so you:
  - Merge the shelved content from the personal stream back into the work area.
  - Resolve any conflicts between the changes made for the bug fix and new feature.

Shelving Changes

1 In the Synchronize view right-click the stream that you want to shelve and select **Team** | **Shelve**. The Shelve wizard appears.

2 Enter a name for the new personal stream, the default is:

```
<Username>_<ID of the parent stream>
```

3 (Optional) Modify the default description of the personal stream.

4 Enter a unique branch name, the default is:

```
<Username>_<ID of the parent stream>
```

5 After the shelving operation is complete, by default the work area is reset to the latest repository content and the local changes are discarded. If you do not want to reset the work area unselect **Reset work area changes after shelving**.

6 Optionally add the new personal stream to your list of favorite streams. Click **Next**.

7 On the Shelve Changes to Repository page select the resources that you want to deliver. Click **Next**.

8 On the Relate Requests page select change requests. Click **Next**.

9 On the Item Attributes page select attributes for the revisions your are delivering.

10 Click **Finish**. If the Update wizard appears you have the option to reset the work area. The stream to be used for the update is the same one that was used to shelve the changes. In the Advanced section the following options are selected by default:

- **Reset work area changes to repository versions and paths**
- **Delete locally added files** (such as artifacts added by a local build process)

Complete the Update wizard, for details see page 38.

Restoring Shelved Changes

To restore shelved changes you merge a personal stream back into a work area. The source stream is the shelved personal stream and the target stream is the stream you are merging into. When restoring you can:

- Select which content to restore.
- Restore to a new, or clean, work area by creating a new workspace and adding stream content to it.
- Restore to a work area associated with a different stream if, for example, the stream used to shelve a work area is locked.

NOTE The process for restoring shelved changes is the same for merging streams.

1 Do one of the following:

- In Package Explorer right-click a stream or folder and select **Team** | **Merge**. The Merge wizard opens. Select **Merge streams or their folders** and click **Next**.

  To select the shelved personal stream do one of the following:

- Select the **Find** option, click **Find**, and use the Select Stream wizard to specify the stream.
- Select the **Browse** option and select a stream from the Repository Projects tree.

Click **Next**.

- In Dimensions CM Explorer right-click the shelved personal stream and select **Merge**. The Merge wizard opens.

2 The **Merge changes from this stream** box displays the ID of the shelved personal stream that will be the source for this merge.

3 A table lists all the Eclipse IDE projects currently in your workspace. These are the projects that you will merge into. To change the scope of the merge do the following:

a Click **Select**.

b Select a projects group from your workspace.

c Click Next.

d Select the workspace projects that you want to merge into.

e Click **Finish**.

4 To interactively (manually) verify the merge results before applying them to the work area, select **Perform an interactive merge**.

5 (Optional) Click **Advanced**.

6 Select these options where applicable:

- Apply the repository date and time to the files in the merge.
- Automatically merge files whose content does not conflict.
- Show a summary of the results of the operation (only available for non-interactive merges).
- Enable logging. Enter the folder path where the log file will be saved or click Browse and select a folder.

Click **Next**.

7 Select a version of the source stream.

8 To run the merge click **Finish**. The results of the merge are displayed in the Synchronize view. Use the view to compare revisions, resolve conflicts, and synchronize with the repository.

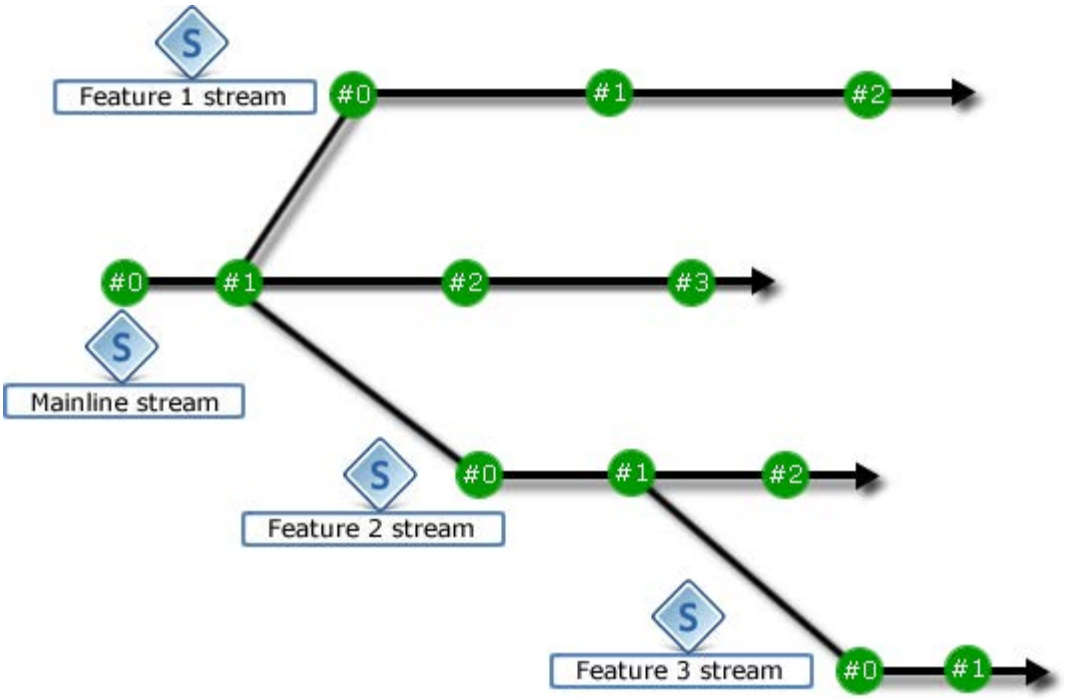
Working with Changesets

About Changesets

A changeset is a logical grouping of changes that is created automatically every time that you deliver changes to a stream or project in a repository. Each changeset creates a new version of a stream or project. Changesets give insight into the development activity in

your streams and enable you to easily identify changes. They also reduce the complexity of parallel development by making it easy to manage sets of changes. PulseUno provides you with an intuitive visualization of changes throughout your development process.

The diagram below shows multiple streams where each circle represents a new stream version when a delivery has been made. Changeset version #0 (zero) is empty and represents the initial state of a stream or project.



The Changesets view enables you to view all of the changes that have been made to a stream or project since it was created and includes the following information:

- The stream version created by the new changeset, for example: `[123]`
- The date and time the stream version was created, for example: `11/22/2013 15:20:13`
- The name of the user who made the delivery, the type of delivery, and any comment entered by the user at the time of delivery, for example: `DELIVER by ANON:"Modified"`

Each item in a changeset includes the following information:

- The type of change, for example, *Modification*.
- The folder path.
- The item revision.
- Any related change requests.

Viewing Changesets

1 To display all the changesets in a stream or project, do one of the following:

- In Package Explorer right-click the root of a stream or project, or a subfolder, and select **Team** | **Changesets**.
- In Dimensions CM Explorer right-click a stream or project and select **Changesets**.

The Changesets view is displayed in a new tab.

2 To filter the view by a date range do one or both of the following:

- To display entries from a specified date, select the **From** check box and select the date.
- To display entries up to a specified date, select the **To** check box and select the date.

3 To filter the view by an attribute:

- From the **Filter** list select an attribute type.
- Enter a value in the box.

For example, select *Request(s)* as a filtering attribute and in the Filter box enter a request type, such as ENH.

Inspecting Changeset Health

Changeset health is displayed visually using a combination of:

- Color: the opinion of the experts that ran on the changeset.
- Icon shape: the review state of a changeset

You can use experts in PulseUno to:

- Perform builds
- Capture built artifacts
- Deploy assets
- Examine source code and artifacts, report findings, and provide an opinion
- Perform static analysis

You can configure experts and tools to run in a sequence of steps called a chain. You can also configure changesets to automatically create reviews and run chains of experts. For details about using PulseUno experts see the PulseUno help.

Examples of changeset health:

The changeset is healthy and the review was successful.

The changeset is not healthy and the review failed.

The changeset is unstable.

No changeset was delivered and no review is available.

The changeset was aborted and there is no review.

1 Do one of the following:

- In Dimensions CM Explorer right click a stream or project and select Changesets. The Changeset views is displayed.
- In Package Explore, right click a Java project or a subfolder, and select Team | Changesets.

The icon on the left indicates the health of each changeset.

2 To open the review associated with a changeset, click the health icon. The review opens in a new view.

3 To view a changeset's details, right-click and select Open Details. The changeset opens in a new view.

NOTE

- To use PulseUno functionality it should be enabled in your database.
- To hide the changeset health status icon, from the Configure menu on the Changesets view select Configure columns and uncheck Show review status.
- Changeset view has a Review column.

Opening Reviews

To display the review associated with a changeset, in the Changesets view, click the health icon.

Viewing Changeset Details

You can view changeset details including its associated review and contents.

1 Open the Changeset view.

2 Right-click a changeset and select **Open Details**.

The changesets details are displayed in a new tab.

See also: PulseUno help

Comparing Changeset Items

You can compare an item in a changeset item with the previous revision of the same item.

1. Open the Changeset view.
2. Right-click an item and select **Compare**. Both revisions open for comparison in a new tab.

Displaying the History of a Changeset Item

You can display the revision history of a changeset item.

1. Open the Changeset view.
2. Right-click an item revision and select **History**. The item revision history opens in a new tab. You can right-click a revision and perform additional operations.

Browsing a Changeset Item Revision

You can browse the content of a changeset item.

1. Open the Changeset view.
2. Right-click an item revision and select **Browse**. The revision opens in a new tab.

Browsing Requests

You can browse the change request that is related to a changeset item.

1. Open the Changeset view.
2. Right-click a item that has a related change request and select **Open Request**. The request opens in a new tab.

Customizing Changeset Columns

You can customize the columns that are displayed in the Changeset view and change their order.

1. Open the Changeset view.
2. Click **View Menu** in the top right corner and select **Configure Columns**.
3. To show or hide a column select it and click **Show** or **Hide**.
4. To change the position of a column, in the **Shown** box select it and click **Up** or **Down**.
5. Click **OK**.

Opening a Changesets Graph

You can visualize your streams of development and changesets in the changesets graph.

Do one of the following:

- In Package Explorer right-click a stream or project and select **Team** | **Open Changeset Graph**.
- In Dimensions CM Explorer right-click a stream or project and select **Open Changeset Graph**.

The changesets graph is displayed in an embedded browser in a new tab.

TIP To configure the browser and how it appears go to Window | Preferences | General | Web Browser section.

Working with Files

Operations in Compatible Work Areas

Using "compatible" work areas you can use different Dimensions clients to perform source control operations, such as file check in and check out. You can also use the other clients to deliver changes and update local work areas.

If you perform source control operations outside eclipse that change the local files and metadata or change the repository, you must refresh the Eclipse display and internal caches to reflect this. Use the **Team | Refresh Status** menu option at the Eclipse project level to ensure that the status is current.

Dimensions for Eclipse uses project specific upload rules when delivering new files. These rules specify the item type, item format and owning design part for new items. These rules are created whenever an Eclipse project is shared with Dimensions CM. Whenever new files are added within the scope of a shared Eclipse project, the original upload rules for that project will be used, regardless which client performs the action

Dimensions for Eclipse does not consider derived resources as candidates for source control. However, when using other Dimensions clients, these files may be considered as candidates for source control. The default exclusions filter on the Dimensions Synchronize wizard excludes common java derived artifacts such as class, jar and war files from source control. Take care to exclude these files if you have changed the Synchronize wizard filter, or if you are using the command line interface.

NOTE: If you are performing cross-stream merges including deletions and file renames/ moves, then this operation and the following delivery back to Dimensions CM must be performed completely in either Eclipse or the other Dimensions client.

NOTE: If you are using Dimensions projects, it is strongly recommended that you use the Dimensions for Eclipse to perform check-in operations. Dimensions for Eclipse maintains and expects a single line of descent in Dimensions projects and this is not supported in other Dimensions CM clients. If you get branches in the pedigree of an item in a shared Eclipse project, you will need to resolve this to maintain a single tip revision within the Dimensions project.

Checking Out Files

When you check out a file, a writable copy of the latest revision is placed in the work file location and assigned the next revision number.

You do not need to check out individual files if you are working with Dimensions streams, or if your organization follows an optimistic locking model.

To check out a previous version of a file, you must select the specific revision using the file history.

NOTE If you are logged into your system as the root user, files you have checked out will not be set to writable. They will be set to read-only. You must manually make the files writable before you can work on them.

To check out files:

1 In Eclipse, right-click the files and select Team | Checkout.

NOTE The project file contains information about the project. Even if you do not explicitly check out this file, Eclipse may silently make this file writable to note the changes to the project.

The **Checkout** dialog box appears with a list of selected files to be checked out.

NOTE You must share the project which contains the file to activate the checkout command.

2 On the Checkout dialog box, do any of the following and click **Next**:

- Select or deselect which files will be checked out.
- Select from available options on **Options** tab. For example, change the value to *Prompt* for *Overwrite modified files* to be prompted if the file in your local workspace is different than the file being retrieved from the Dimensions CM repository.

3 If you are prompted to relate requests to the files that you are checking out, select the requests from the My Inbox or My Working List tabs. Click **Next**.

4 On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the check out operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

5 On the **New Item Revision Attributes** panel, set the Dimensions CM attributes for the items that you are checking out.

The attributes are grouped by type. You can copy attributes by right-clicking and selecting from the resulting menu.

6 Click **Finish**.

On successful completion, a check mark ( ) appears next to each file icon to indicate that the files are checked out.

Locking and Unlocking Files in Streams

If you are working with Dimensions Streams, you can lock files to prevent other users from delivering changes while you are working the files. This is particularly useful for file types that cannot be easily merged, such as binary graphics files.

To lock files:

1. In Eclipse, right-click the files and select Team | Lock. The Lock dialog box appears with the list of files to lock. You can optionally de-select files to lock from here.
2. Click **Finish**. The locked files appear with a padlock icon. Note that any files with brown padlock icons are currently locked by other users.

To unlock files:

1. In Eclipse, right-click the files and select Team | Unlock. The Unlock dialog box appears with the list of files to unlock. You can optionally de-select files to unlock from here.
2. Flick **Finish**.

Undoing Check Out

When you undo a check out, the lock and the revision number that was created during check out is released. No changes are checked into the Dimensions CM repository.

To undo a check out:

1. Right-click the checked out files and select Team | Undo Checkout.

   The Undo Checkout dialog box appears with a list of selected files.
2. On the Checkout panel, do any of the following:
   - Select or deselect which files to act on.
   - Select from available options on the **Options** tab. For example, change the value to *Replace with Latest Revision* for *Local contents* to overwrite the file in the local workspace with the latest revision in the Dimensions CM repository.
3. Click **Finish**.

Checking In Files

Complete these steps to check in, or deliver, changes to the Dimensions repository.

To check in files:

1. In Dimensions CM Explorer, right-click the files to check-in and select Team | Checkin.

   The Deliver Changes dialog box appears with a list of selected files.
2. Do any of the following:
   - Select or deselect specific files to check in.
   - Enter a description of the files on the **Comment** tab.

- **NOTE** If your item type rules require a comment, the **Next** and **Finish** buttons will be disabled until you add a comment.
- Select from the available options on the **Options** tab:
  - Set **Checkout posted changes** to **Yes** to check out the files that you are checking in, once check-in is complete. If you choose this option, when check-in is complete, the revision that you are checking in will then be checked back out to you.
  - Set **Make files read-only** to **Yes** to make the local copies of the files read-only after check-in is complete.
  - Set **Overwrite conflicting changes** to **Yes** to overwrite any conflicting changes in the repository with local content. If you set the option to **No**, the checkin will fail if there are any conflicts, and you will be prompted to merge the files using the synchronize view.
  - Set **Undo Checkout if unmodified** to **Yes** to unlock the file without checking in a new revision, if the local file is not different from the latest revision in the repository.
- Compare the file that you are checking in with the revision that resides in the repository on the **Differences** tab.

3 On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the check in operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

4 On the **Item Attributes** panel, set the Dimensions CM attributes for the items that you are checking in.

5 Click **Finish**.

On successful completion, the check mark is removed from each file icon to indicate that the files are checked in.

Accessing a Previous Revision

You can check out a previous revision of a file, rather than the latest revision, using the Dimensions for Eclipse integration.

To access a previous revision:

1 Right-click the file and choose Team | Show History.

2 Select the revision you want from the Item Revision History view.

Adding New Files to Source Control

Once you have connected an Eclipse project to source control, you can add new files to source control at any time.

If you are using Dimensions streams, use the synchronization feature to add new files to source control.

To add new files to an Eclipse project that is already under source control:

1. In Eclipse, right-click the files and select Team | Add to Dimensions.

   The **Deliver Changes** dialog box appears with a list of selected files to be added.

2. On the Deliver Changes panel, do any of the following and click Next:
   - Select or deselect which files to add to source control.
   - Enter a description of the files on the **Comment** tab.
   - Select from the available options on the **Options** tab:
     - Set **Checkout posted changes** to **Yes** to check out the files that you are checking in.
     - Set **Make files read-only** to **Yes** to make the local copies of the files read-only after check-in is complete.
     - Set **Overwrite conflicting changes** to **Yes** to overwrite any conflicting changes in the repository with local content. If you set the option to **No**, the checkin will fail if there are any conflicts, and you will be prompted to merge the files using the synchronize view.
3. If you are prompted to relate requests to the files, select the requests from the **My Inbox** or **My Working List** tab and click **Next**.
4. On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the add operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

   By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.
5. On the **Item Attributes** panel, set the Dimensions CM attributes for the items that you are adding to control.
6. Click **Finish**.

   On successful completion, a gold cylinder is added to the icon of each item to indicate that the item is under source control.

Removing Files from Source Control

When you remove a file from source control, the file is removed from the Dimensions CM project. You will no longer be able to access previous revisions of the file from this project. Only the latest revision will remain in your local area (if you do not opt to delete it).

The file will still exist in the $GENERIC:$GLOBAL project (the global project) should you need to restore it later.

To remove files from source control:

1. In Eclipse, right-click the files under Repository Projects. A pop-up menu appears.
2. Select **Team | Remove from Dimensions**.

   The Remove From Control dialog box appears with a list of files to be removed.

3 On the Remove from Control panel, do any of the following:

- Select or deselect which files will be removed from control.
- Select from available options on the **Options** tab. For example, change the value to Yes for *Delete local files* to delete the files in local workspace after removing the files from Dimensions CM control.

4 On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the remove operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

5 Click **Finish**.

Viewing Files in a Different Perspective

You can see which files are under Dimensions CM control from other perspectives within Eclipse, such as the Java or the Resource perspective.

If you have enabled the Dimensions CM indicator (default), the files are marked with icons indicating that they are under control and if the local versions differ from the one in the repository. The displayed file names also have the version number in parenthesis appended at the end.

NOTE The file decorations can be enabled from the Eclipse preferences. Select Windows | Preferences, then select General | Appearance | Label Decorations. Select the **Dimensions** check box to enable Dimensions file decorations.

You can access most of the source control options by right-clicking on the file and selecting the **Team** menu.

Viewing Properties for a File

The Dimensions for Eclipse integration allows you to access file attributes, history, and relationships information through a single dialog. You can also use this dialog to perform source control operations upon specific revisions of files.

To view source control information:

1 In Eclipse, right-click the file or project and select **Team | Open Item Properties**.

The item properties appear in the **Item Revision** editor.

2 You can browse the different properties for the item on the different tabs:

- **Details**—Displays general details about the file such as revision number, status, stage, and lock status.
- **Attributes**—Displays the item attributes for each of the different roles.
- **Relationships**—Displays related objects such as design part, requests, baselines, and other items.

- **History**—Displays the item history for the file, as compared to the revision history which displays with the *Team* | *Show History* command.
- **Users and Roles**—Displays the Role Assignments and Request Pending For information for the item.
- **Privileges**—Displays the item privileges for the selected user.

Comparing a File with a Previous Revision

To compare the current local file with a previous revision, right-click the previous revision on the Item Revision History view and choose **Compare with local**.

The files display in the Eclipse compare view.

Merging a Local File with a Repository File

You may need to merge two revisions of a file if the current revision in the repository is newer than your local revision.This scenario occurs in a concurrent development paradigm, where multiple developers are working on the same project.

For example, you get revision 4 of the file from the Dimensions CM repository. You mark the file as local and update the file without checking it out. You decide that you want to commit these changes to the repository but you find that someone has checked in revision 5 of the file while you were performing the updates. You will need to merge your locally changed revision with the new revision in the repository.

NOTE Your local revision of the file may be either checked out or in local mode. The Dimensions for Eclipse integration supports the merging of either type of file with the more recent tip.

If you were to synchronize your project, you would see that the file has an arrow pointing both directions, meaning that the file has changed in both locations.

The process involves merging the local and repository files into your local directory, and then committing the new file into the repository. Note that merges that resolve concurrent conflicts never result in merge transitions stored in the pedigree.

To perform this process from the synchronize view:

1. First right-click the file in the Project Explorer and select Team | Synchronize with Repository. The Synchronize view appears.
2. Merge the two files into your local file.
   a. Right-click on the file and select **Open in Compare** editor.
   b. Move the changes to keep into the local version of the file.
   c. Save the merged file.
3. Mark the local file as merged by right-clicking on the file and selected **Mark as Merged**.
4. Commit the merged file into the repository.

Merging a Local File with a File from Another Project

You may need to merge two versions of a file found in different projects in the repository. This scenario occurs frequently within a parallel development paradigm in which two releases of an application are being developed in parallel from a single baseline.

For example, after release 2.0 of a product is released and the project is baselined, two projects, release 2.1 and release 3.0, start from the baseline at the same time. A developer may be editing a file in project release_2.1, while parallel development is occurring on the same file in release 3.0 of the product under project release_3.0 in the repository. It is decided to merge the changes from release 3.0 into the changes made for release 2.1. This means that the developer has to merge the file from project release_2.1 with the similar file from project release_3.0.

For parallel merge, transitions are not stored in the pedigree when the local file is checked out. It is recommended to perform merge in local mode.

The process involves merging the two files into your local directory, and then committing the new file into the repository.

To merge your file with a similar file from another Dimensions CM project:

1 Right-click on the file and select **Team | Merge**.

The Merge dialog box opens.

2 Select the Dimensions CM project or baseline containing the file that you want to merge with and click **Next**.

3 Select the common ancestor baseline (this is the most recent baseline that both merge participants are derived from) and click **Next**.

4 Click **Finish** on the Merge Summary panel. The Synchronize view appears.

5 On the Synchronize view, merge the file and then mark the local file as merged by right-clicking on the file and selected **Mark as Merged**.

6 Commit the merged file into the repository.

Viewing the History of a File

The history of a file shows the previous versions of a file in the Dimensions CM repository. The history displays in the Dimensions CM History view.

To display the revision history of the file, right-click the file and choose **Team | Show History**.

From this view, you can perform the following actions by right-clicking on a previous revision and selecting the command from the context menu:

- Check out a previous version of a file (**Check Out**). Note this option is only available if file is not checked out locally.
- Get the contents of a previous version of a file (**Get Contents**).
- Place a previous version of a file into your local area without performing a check out (**Get Revision**) Note this option is only available if file is not checked out locally.
- Action a previous revision of a file (**Action**).

- View the contents of a previous revision (**Open**).
- View the properties of a previous version (**Open Item Properties**).
- Compare the current local file with a previous revision of the file (**Compare with local**).
- Refresh the list by clicking the **Refresh** icon.

Comparing a File with the Local History

You can compare a work file with a local history of the changes made to that work file. A new entry is made in the local history each time that you save changes to a file.

NOTE The compare utility is provided by Eclipse, and is only described briefly here. For more detailed information, see the online help provided with Eclipse.

TIP To configure how many entries are retained and for how long, select Window | Preferences then select **Local History** from under **Workspace**.

To compare with local history:

1. In Eclipse, right-click the file and select Compare With | Local History.

   The Compare with Local History dialog box appears.

2. Select a local history entry from the Local History of *filename* pane.
3. Use the **Select Next Change** and **Select Previous Change** buttons to step through the changes.
4. Click **OK** to exit the compare.

Replacing File with Local History

You can replace a workfile with an entry from the local history of changes made to that workfile. A new entry is made in the local history each time you save changes to a file.

NOTE The replace utility is provided by Eclipse, and is only described briefly here. For more detailed information, see the online help provided with Eclipse.

TIP To configure how many entries are retained and for how long, select Window | Preferences then select **Local History** from under **Workbench**.

To replace with local history:

1. In Dimensions CM Explorer, right-click the file to replace and select Replace With | Local History. The file must be checked out or writable.

   The Replace from Local History dialog box appears.

2. Select a local history entry from the Local History of *filename* pane.
3. Use the **Select Next Change** and **Select Previous Change** buttons to step through the changes.

4 Do one of the following:

- To replace the work file with the selected history entry, click the **Replace** button.
- To close the dialog box without replacing the work file, click the **Cancel** button.

Using Local Mode with Optimistic Locking

When you are working with Dimensions projects, you can edit files without checking them out from source control by working on them in local mode. Local mode can be enabled for individual files, unlike offline mode which affects the entire workspace.

NOTE
You can choose to enable local mode at the project level, which will place all files contained by the project into local mode. The project itself cannot be in local mode.

You can use local mode with a pessimistic locking environment to work on files without checking them out, and directly check in the updated files. Local mode does not apply to Dimensions streams.

If you place files in local mode and fetch newer revisions of the files from the repository, the newer revisions are not placed into local mode automatically.

Putting Files into Local Mode

To put files into local mode:

1 Do one of the following:

- Select the projects or files that you want to work on in local mode, and select Team | Local Mode from the context menu.
- Open a file and attempt to edit it in the editor. You are prompted to either check out the file or make it local.

  The default choice is **Make Local**. To avoid this dialog in the future and always make the file local, you can select **Don't ask me again**.

  NOTE
  You can change your default setting in the Preferences tab: **Window | Preferences | Team | Dimensions | Version Management | File Modification**

The local mode icon appears on the icons of the selected files.

2 If your business processes support an optimistic locking approach, check in the files after making the edits in local mode by selecting **Team | Checkin.**

The new revision of the file is placed at the tip (latest revision) of the default working branch.

Reverting Local Mode Files to Controlled Mode

NOTE
If your business processes support an optimistic locking approach, you can choose to directly check in the files after making the edits in local mode, instead of using the revert to control command.

To revert files to controlled mode:

1. In the Package Explorer or Navigator, select the local mode projects or files you want to revert to controlled mode and right-click. A pop-up menu appears.
2. Select Team | Revert Controlled.

   The file is marked as read-only.
3. Use the **Team | Synchronize with Repository** command to commit your changes into the repository.

Enabling File Content Encodings

Eclipse supports a number of file encodings. By default these are not passed on to Dimensions CM. If content encoding support when adding files to the repository is required, edit the file **plugin.properties** in the following directory:

```
<Dimensions_Install>\Integrations\RichEclipse3.x\eclipse\plugins\
    com.serena.eclipse.dimensions.team.core_14.3.0.0
```

By adding the following line to the file:

```
use_content_encoding=true
```

With this set, the Eclipse Content Encoding set on the Eclipse resource will be passed onto Dimensions CM. If Eclipse content encoding is changed between revisions or on fetching files to a clean workspace, it must be changed in the Eclipse File, Folder or Workspace properties to allow Eclipse to recognize and display files correctly.

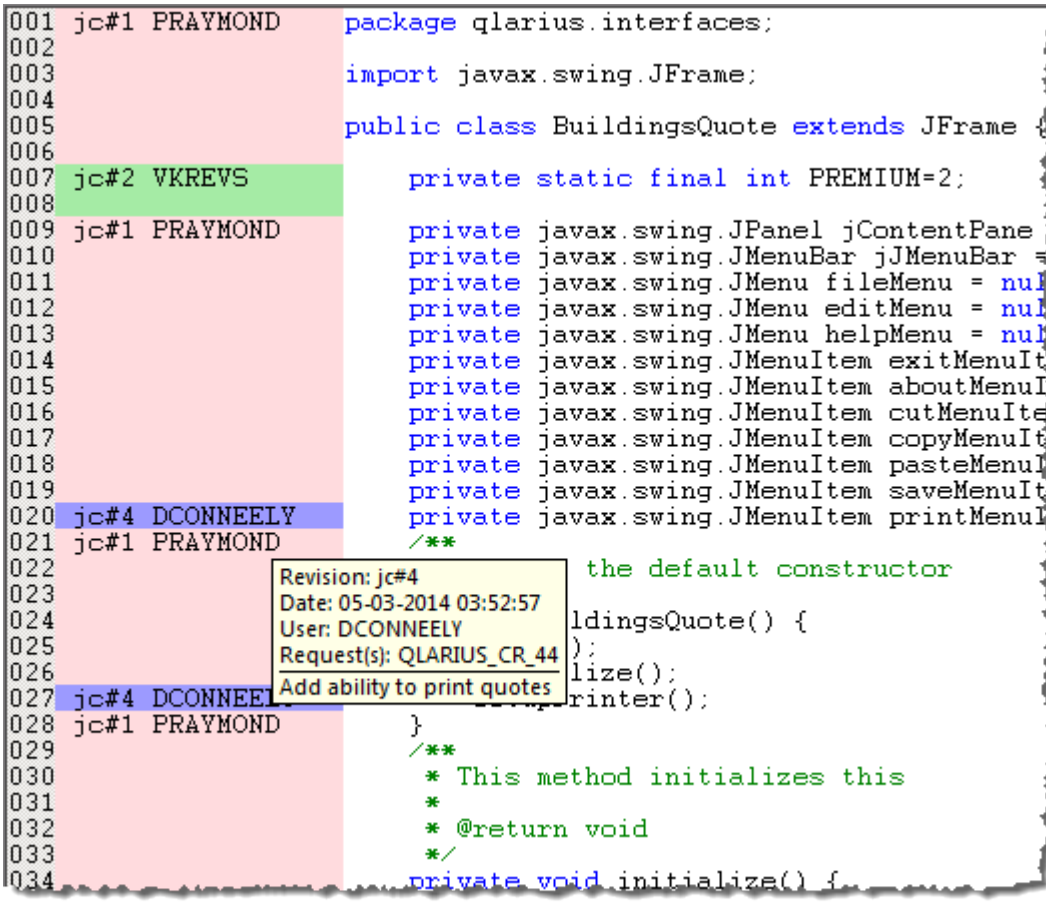
Viewing File Annotations

About Annotations

The Annotations view annotates the lines of code in your source files. Annotations make it easier to find when a change was introduced. You can also see who made a change and why. Each annotation displays the item revision and the name of the user who delivered the change. Hover over an annotation to view additional information about an item revision such as:

- The date and time of the delivery.
- Its related change requests.
- Any comments added by the user

You can also use the ANNOTATE command in the DMCLI command-line client to list file annotations.



Opening the Annotations View

The Annotations view enables you to see who has modified blocks of code and why. Each user's deliveries are highlighted with the same color.

To open the Annotations view:

1. In Package Explorer right-click a file and select **Team** | **Show Annotations**.
2. In the left pane place the cursor over an item revision. A tooltip displays the following information:
   - The item revision.
   - The user who made the modification.
   - The date the modification was made.
   - Any related change request.
   - Comments added by the user when the item revision was delivered.

   To keep the tooltip in focus press F2.

Viewing the Revision History of an Item

You view the revision history of an item revision.

To view the revision history of an item:

1. Open the Annotations view for an item.
2. Right-click an item revision and select **Show History.** The item revision history opens in a new tab.

Browsing an Item Revision

You can browse the contents of a specific item revision.

To browse an item revision:

1. Open the Annotations view for an item.
2. Right-click an annotation and select **Browse Revision**. The item revision opens in a new tab.

Creating Baselines

In Dimensions CM, a baseline is a snapshot of the project at a particular time. You can capture and record all of the tip versions of items in a project by saving them as a baseline. Baselines ensure that the items included in the baseline can be reliably recreated in the future. For example, create a a baseline before starting a maintenance cycle to allow you the ability to return to the original set of released files.

For more information on baselines in Dimensions CM, see the *Dimensions CM Process Configuration Guide*.

You can create the following types of baseline from any project or stream in the repositories you are connected to:

- A baseline that uses a specific baseline template that determines which revisions to include, and that is scoped by design part
- A design baseline that represents the current product design structure, or a part of it
- A tip baseline that includes the latest revisions of all files in a project
- A revised baseline, which is a baseline that is derived from an existing baseline by adding or removing item revisions that are related to requests

Creating a Baseline

Follow these steps to create a baseline.

To create a baseline:

1 Do either of the following to invoke the Create Baseline wizard:

- Right-click a node from an open connection on the Dimensions CM Explorer and select Team | Create Baseline. If you right-click on a project or stream, relevant information such as product and project will be pre-populated.
- Right-click a controlled project in the Eclipse Navigator pane and select New | Baseline.

2 Choose the product containing the project or stream, and then choose the project or stream.

3 Select the type of baseline from the **Type** list.

4 Enter an ID string for the baseline.

5 From **Create from project**, choose the project or stream from which to create the baseline, or click **Browse** to select it.

6 Click **Next**.

7 On the Select Baseline Template and Design Part screen, select a baseline template from the **Template** list. The baseline template determines which requests or items to include in the baseline. To create a design baseline, do not select a template.

8 Enter a design part ID in the **Top Part** field, or click the **Browse** button to search for a design part. This field identifies the top level design part in the design part hierarchy from which you will baseline items or requests. You can include any number of levels of children of this design part using the **Part levels** options.

9 The **Part levels** options allow you to choose the levels in the design part hierarchy from which items will be included, starting from the parent design part you entered in the **Top Part** field. To include items from all design parts under the top level design part, select **All**. To specify the number of levels from which to include items, select the **From top to Level** option and enter or click to choose a number. For example, if you select **From top to Level** and enter 3, then items will be included from the top three levels of the design part hierarchy, starting with the design part you entered in the **Top Part** field.

10 Click **Next**.

11 On the Select Requests screen, enter or browse to select any requests that you want to relate to this baseline. If you selected a request baseline template, the baseline will include any items that have an "in response to" relationship to the requests you enter here. If you selected an item baseline template, the requests you enter here will just be related to the baseline but have no effect on the content of the baseline.

12 Click **Next**.

13 On the Baseline Attributes screen, enter values for user defined attributes, and click **Next**. You must enter values for required attributes in order to create the baseline.

14 Click **Finish** to create the baseline.

Creating a Tip Baseline

To create a tip baseline:

1 In the Dimensions CM Explorer, right-click the project and choose **New | Tip Baseline**, or right-click the project in the Eclipse Explorer and choose Team | New Tip Baseline. The Create Tip Baseline dialog appears.

2 On the Create Baseline panel, select the Dimensions CM Product, Type, Baseline ID, From Project and IDE Project., then click **Next**.

NOTE
The Baseline ID is auto generated as project_blx, where x is a number to make the baseline unique.

3 On the Select Requests screen, enter or browse to select any requests that you want to relate to this baseline.

4 On the Baseline Attributes screen, enter the attributes for the new baseline.

5 Click **Finish**.

Creating a Revised Baseline

To create a revised baseline:

1 In the **Dimensions CM Explorer**, right-click the baseline for which you want to create a revised baseline, and select New | Revised Baseline.

2 The product and project are pre-populated based on your initial baseline selection.

3 Select a baseline type from the **Type** list.

4 The ID string for the revised baseline is auto-generated.

5 Click **Next**.

6 In the **Update baseline using** field, list the requests related to item revisions you want to add to the baseline. The item revisions must have an *In Response To* relationship to the requests. You can enter one or more IDs separated by a comma, or click **Browse**.

7 In the **Remove from baseline using** field, list the requests related to item revisions that you want to remove from the baseline. The item revisions must have an *Affected* relationship to the requests. You can enter one or more IDs separated by a comma, or click **Browse**.
NOTE You must have at least one request in either the **Update baseline using** or **Remove baseline using** field.

8 Select the **Traverse request relationships** check box to search for item revisions included in requests related to the listed requests.

9 In the **Scope using** field, enter the ID of the project or stream from which you want to include item revisions. Click **Browse** to find a project or stream.

10 Click **Next**.

11 On the Baseline Attributes screen, enter values for user defined attributes, and click **Next**. You must enter values for required attributes in order to create the baseline.

12 Click **Finish** to create the revised baseline.

Refactoring Projects and Files

Note About Refactoring

Refactoring includes moving or renaming projects or project elements. For best results, keep the following in mind when performing refactoring operations:

- Before refactoring, all users should check in their changes.
- After refactoring, all users should get the updated project from source control.

Refactoring includes renaming files within a project and moving a file between two different projects. The Dimensions for Eclipse integration retains the file history when you move a file between projects using the refactoring option.

When performing a move between projects as part of refactoring, it is strongly recommended that you choose the option to set the files to be updated as part of the refactoring in "Local Mode". This is not necessary when working with Dimensions streams.

If the files are checked out, then there is a conflict with the removal of the source file from source control as part of the cross-project move operation. The checked out revision will not be removed from source control, and you will receive a message to this effect in the console. The checked-out revision will remain in Dimensions CM and will be picked up by subsequent project structure synchronization operations. These revisions must have the checkout canceled and be removed from the source project using the desktop client.

A particular case of this is moving a file to a different package in the second project. The Eclipse java tooling identifies that it needs to update the file with the new package name and initiates a checkout operation.

If the file is checked out, then the subsequent attempt to remove it from source control will not remove the checked out revision. Thus the recommendation to use local mode. This will make the file writable and the java refactoring code will be able to update the file as part of the move to the new package in the target project. As it is not checked out, the remove will succeed.

The same will apply if files are checked out by another user prior to performing the refactoring. The revision checked out to the other user will not be removed and a warning will be issued in the console. These revisions must have the checkout canceled and be removed from the source project using the desktop client as the appropriate user.

We recommend using the "Team | Refresh Project Status" feature to check that files which will be updated and moved are not checked out by other users (The "Lock" icon or layered "Lock" icon) or are not checked out by you and other users (The layered "Check" icon) prior to performing the refactoring.

Renaming Projects or Project Elements

To rename a project or project element:

1 Before renaming projects or project elements, all users should check in their changes.

2 Select the item that you want to rename.

3 Depending on your perspective and the type of item:

- Right-click and choose **Rename.** Type the new name of your resource.
- If you are renaming a single Eclipse project, choose **Refactor | Rename**.

  If the file is not already checked out or made local, you are prompted to check it out or make it local. Select **Make Local** and click **OK**.

The new name has been applied to the local file, and both the new name and old names are shown in the perspective.

4 If you are renaming a non-project item or an Eclipse project that is shared in a container. Commit the change to the repository by right-clicking the file and selecting **Team | Commit Move**.

The **Commit Move** dialog appears.

a If necessary, deselect any items that you do not want to commit to the repository and change the commit options. For example, if you select *Yes* for **Convert to outgoing additions**, a new file will be created rather than moving the existing one.

b On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the commit operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types. By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

c Click the **Finish** button. The file is placed into local mode.

5 To perform a synchronization to deliver the changes to the repository, do one of the following:

- Projects: **Team | Synchronize with repository**
- Streams: **Team | Deliver**

After you have performed the synchronization the name is changed in the Dimensions CM repository.

6 All users should now get the updated project or project element from source control.

Moving items in a Project

To move an item within a project:

1 Before moving the item, check it in.

2 Select the item that you want to move.

3 Depending on your perspective:

- Right-click and choose **Move.** Select the new location for your resource.
- Choose **Refactor | Move**. The Move dialog appears.

  Select the new location for the file and the desired options.

  If the file is not already checked out or made local, you are prompted to check it out or make it local. Select **Make Local** and click **OK**.

The file is moved to the new location. The old location is shown alongside the file name.

4 Commit the changes for the moved file to the repository by right-clicking the moved item and choosing **Team | Commit Move**.

The **Commit Move** dialog appears.

5 Deselect any items that you do not want to commit to the repository and change the commit options as needed. For example, if you select *Yes* for **Convert to outgoing additions**, new files will be created and the originals deleted, instead of moving the existing ones. If you are prompted to relate requests, choose the requests and complete the Commit Move dialog box.

6 On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the commit operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

7 Synchronize the project by selecting the project and choosing **Team | Synchronize with Repository**. For example, if other elements were automatically updated with new references to the file location, you must check in the updated elements.

8 All users should now get the updated project or project element from source control.

Moving items Between Projects

To move an item between projects:

1 Before moving the item, check it in.

2 Select the item that you want to move.

3 Depending on your perspective:

- Right-click and choose **Move.** Type the new name of your resource.
- Right-click the file and choose **Refactor | Move**. The Move dialog appears.

  Select the new location for the file and the desired options.

  If the file is not already checked out or made local, you are prompted to check it out or make it local. Select **Make Local** and click **OK**.

The file is moved to the new location. The old project and location are shown alongside the file name.

4 Commit the changes for both the old and the new projects to the repository.

a Under the new project, right-click the moved item and choose **Team | Commit Move**.

The **Commit Move** dialog appears.

b Deselect any items that you do not want to commit to the repository and change the commit options as needed. For example, if you select *Yes* for **Convert to outgoing additions**, new files will be created and the originals deleted, instead of moving the existing ones.

c On the **Relate Requests** panel, select the Dimensions CM requests to relate to the item during the commit operation. The items are grouped by type, allowing you to easily relate requests to multiple items with similar types.

By default, only those requests that can be used are displayed. If you choose the **Show All** option, the unavailable requests appear in italics and the **Details** column provides information on why they cannot be used.

d Synchronize the old project with the repository by right-clicking the project and choosing **Team | Synchronize with Repository**.

5 All users should now get the projects from source control.

Using Eclipse Project Containers

What Are Eclipse Project Containers?

An Eclipse project container is a type of Dimensions CM project that can store multiple Eclipse projects. An Eclipse project that is stored within an Eclipse project container is called a *contained project*.

Eclipse project containers are displayed in the Dimensions CM Explorer view. Derived Dimensions CM projects and baselines are displayed within the project container. Derived baselines are grouped together under a common node within the project container.

Why Use Project Containers?

By default, single Eclipse projects are mapped 1 to 1 with individual Dimensions CM projects. The project container option allows you to override this way of working in order to nest any number of Eclipse projects under a common Dimensions CM project.

If you have already defined scripts or processes that operate against a nested hierarchy of Eclipse projects, you can continue to maintain that project structure. You can also use project containers to minimize the number of Dimensions CM projects that you must create and maintain.

When to Use One-to-One Project Mappings?

In some scenarios, one-to-one mappings between your Eclipse and Dimensions CM projects are desirable. Consider using one-to-one project mapping in the following cases:

- If you need to individually baseline each Eclipse project. When you use project containers, you can only baseline all projects contained within the container.
- If you will use the project grouping features in order to work on specific sub-sets of Eclipse projects in one workspace. You can use groups with Eclipse project containers, however you must include all projects in the container. You cannot select specific projects from a container.
- If your teams will work on different Eclipse projects that are separate from each other; the one-to-one mapping enables organizations to choose different combinations of development projects as needed using the project grouping features.

Project Container Setup Overview

At a high level, you set up project containers by completing the following steps. See the following sections for detailed information on each step.

- Installing Dimensions CM to the Dimensions server.
- Installing Dimensions CM to each system with an Eclipse client installed to it.
- Optionally, if you have previously used Dimensions CM Eclipse SCC projects and want to maintain your existing hierarchy, convert those projects to project containers. You can choose to convert them to project containers instead of upgrading them to the single Eclipse project mapping.

- Select a default way of working with Eclipse projects by setting your preferences correctly.
- When necessary, add Eclipse projects to a Dimensions CM project. If the Dimensions CM project does not already contain Eclipse projects, you can choose to convert the project to a project container when you add the Eclipse projects.
- When adding an Eclipse project to a project container, specify a folder offset path within Dimensions. Each Eclipse project must reside in its own unique folder within the Dimensions project. This is to ensure that multiple projects, with potentially identical names, do not reside in the same folder and therefore create conflicts. By default, the location of each Eclipse project in the Dimensions project is based on the originating Eclipse project name (if the project files are located within the Eclipse workspace). However, you may need to specify a path to ensure that the path is unique, or if the project files exist outside of the Eclipse workspace.

Converting Dimensions SCC Projects

In order to use existing Dimensions CM Eclipse SCC projects as project containers, you must first convert them. When you convert SCC projects, you can choose whether to:

- Convert them to project containers, or to single projects that map one-to-one to Eclipse projects.
- Convert all Eclipse SCC projects in a product to containers at one time.
- Convert a specific Eclipse SCC project to a container. When you convert an existing SCC project, the conversion includes all derived baselines and projects.

To convert existing Dimensions SCC projects to either single projects or project containers, follow the instructions in the following document, available from the Support web site:

Dimensions 10.1: Guide to Upgrading Existing SCC Projects in Eclipse

This document explains how to load the conversion integration to Eclipse and convert your existing projects to container projects.

The following summarizes the key steps that you must follow in order to convert your projects.

1 Load the project conversion integration into Eclipse as described in the *Guide to Upgrading Existing SCC Projects in Eclipse*.

2 To convert all Eclipse projects to the new project container model:

a Expand the **Convert Dimensions Eclipse Projects** node and right-click on **Projects**.

b Select **Convert All to Containers** from the context menu. When you choose this option, all projects that have no parent projects are marked as container projects. All projects under these container projects are converted to contained projects.

3 To convert a specific project to a project container:

Under **Convert Dimensions Eclipse Projects** | **Projects**, right-click the project and choose **Convert to Container** from the context menu. When you choose this option, the project is converted to a container project. All child projects are included in the new container project.

Working with Projects in Containers

If you have converted existing SCC projects to project containers or added new Eclipse projects to containers in Dimensions, all users of the Eclipse project must open it in their local workspaces. To do this, they must choose to add the project containers or individual SCC projects within the project containers to their workspaces within Eclipse. If you choose to add a project container, all projects within that container are added to the Eclipse workspace.

Setting the Default Project Type

You can optionally choose to always add Eclipse projects to Dimensions CM as contained projects, by default. You can then optionally override this default when adding Eclipse projects to Dimensions CM.

To set the default project type:

1. Select Window | Preferences.
2. Expand the Team | Dimensions CM node.
3. If you will work with project containers, select the **Show Eclipse Project Containers in Explorer** check-box.
4. To default to creating contained projects when you add Eclipse projects to Dimensions CM, select the **Default project creation in Eclipse Project Container** check-box.

Adding Eclipse Projects to Project Containers

To add Eclipse projects to Dimensions project containers:

1. Right-click an Eclipse project and select Team | Share Project.
2. In the Share dialog box, select Dimensions as the repository, and click the **Next** button.
3. On the Select Connection dialog box, choose either the **Use existing Dimensions connection** or **Create a new Dimensions connection** option, and select the correct connection. You may be prompted to log in.
4. When you are prompted to choose a Dimensions project, choose to either use an existing project or create a new project. To add the Eclipse project to a container, select the **Create in Eclipse Project Container** check-box.
   IMPORTANT! If the project you are adding to is not already a project container, selecting this option will convert it to a container.
5. If you are adding the project to an existing project in Dimensions that has already been set up as a project container, expand the container under the **Eclipse Project Containers** list and select the project.
6. If necessary, modify the project offset path in the **Container Offset** field. This is a unique path within the Dimensions project to the folder that will contain the Eclipse project file. This path includes a folder named after the Eclipse project. For example, if the project is called **Project1** and you add it to the following path in Dimensions: **/dev/projects**

Then the resulting path will be:
/dev/projects/Project1

IMPORTANT! You cannot move and rename files across multiple Eclipse projects that are stored within a Dimensions project container, when the offset paths for the projects have overlapping folders. For example, you cannot move files from one project to another when those projects share a root folder, such as **/dev**. In order to refactor across projects, those projects must not share a root folder in their offset paths. For example, you can refactor two projects with the following offset paths:
/dev/projects
/qa/test

7 You must also choose the target product, project type, ID (or name) and design part that will provide upload rules for the new project.

8 If you are creating a new project in Dimensions, then you must enter project details, such as the product, type, ID, description, and design part.

Using the Project Explorer

When you use project containers, they appear in the Dimensions CM Explorer as **Eclipse Project Containers**. The contained projects are then nested below them. Projects mapped directly to Dimensions projects appear as **Single Eclipse Projects**.

Using Existing Eclipse Projects with Shared Work Areas

You can use Eclipse projects with shared work areas that exist outside of the Eclipse workspace, see page 27. If you have previously set up Eclipse projects with work areas under the Eclipse workspace, or fetch projects or streams containing Eclipse projects with clients other than the Eclipse client, you can still use these projects with shared work areas. Dimensions CM provides an auto-sharing feature that allows you to get existing eclipse projects and then import them into Eclipse.

1 In Eclipse select Window | Preferences.

2 Expand **Team**, then select **Dimensions | Version Management**.

3 Make sure that the **Auto share on import** option is selected, and click **OK**.

4 Using the desktop or command-line client, download the stream or project that contains the Eclipse projects. Download into a work area of your choice, so long as it resides outside of the Eclipse workspace. You can download an entire stream or project at once, or just the subfolder or folders that contain the Eclipse projects you want to use with the new work areas. You must ensure that the Eclipse marker files are included.

5 In Eclipse, select File | Import.

6 On the Import dialog box, under **General**, select **Existing Project into Workspace**.

7 Click **Next**.

8 On the **Import Projects** screen, choose the **Select root directory** option. Enter or browse to select the root directory under which the eclipse projects you want to

import reside. When you import, Eclipse will automatically detect and share the projects under this root location.

9 **IMPORTANT!** Do not select the **Copy projects into workspace** option.

10 Optionally select **Add project to working sets** if you want to add the imported project(s) to the selected working set.

11 Click **Finish**. If prompted, enter Dimensions login credentials.

The Eclipse projects are imported and automatically shared with Dimensions CM. You can now work with them in Eclipse, as well as update the files in the work area from other clients such as the desktop or command-line client.

Chapter 6

Managing Requests

About Requests

You can use requests to plan, track, authorize, and control all work on the product. A request may capture a defect, enhancement, or other work that needs to be completed on items that belong to the product.

The fundamental concept of managing work with requests is the association of requests to files that you are working on. Follow this process to manage your work with requests:

1. If you organization requires request associations for all work, do not start any work that is not related to a request. You may be unable to deliver your changes to the Dimensions repository without associating your files with requests.
2. Before starting work, identify the requests that you will work on and add them to your working list. You can use requests that other users have already created, or you can create new request.
3. Associate the appropriate issues from your working list to files when checking them out, or when checking them in.
4. The requests then store a record of all affected files. This provides detailed tracking history. Furthermore, the requests can be used for other purposes such as creating baselines of revisions that were created in response to them.

The Dimensions for Eclipse integration simplifies the process of administering requests. You can perform the most frequent tasks, such as actioning a request through its lifecycle or delegating the request to a developer. You can action a request to its next lifecycle state, or any other valid state in the request's lifecycle. Normal Dimensions CM rules only allow transitions that are valid. After the next state has been selected, you can choose to delegate the request to a particular role. Requests that you action are placed in the Inbox of the relevant users.

Where a role has multiple potential users, you can identify the specific users that will receive the request. This allows team leaders to monitor and reassign work as necessary to maintain and improve team productivity.

Working with Dimensions CM Requests

Pre-requisites

In order to work with Dimensions CM requests, Dimensions must be set as the default issue defect management provider. Your Dimensions administrator must set this using the Dimensions Administration Console.

NOTE You must restart your Eclipse IDE before you can work with Dimensions requests.

Displaying Requests

The Requests view displays all requests the requests in a stream or project. You can:

- Display the requests as a flat list of grouped by streams and projects.
- View the reviews associated with a request.

- Customize the columns.
- Apply a filter to display specific request types.

1 Do one of the following:

- In Dimensions CM Explorer, right click an open connection and select Show Requests.
- From the Dimensions menu select Show Requests.

The Requests view is displayed.

2 To browse a request, double click it.

3 To open the reviews associated with a request, right click, and select Open Review.

4 Right click a request to view other operations you can perform on it.

Creating Custom Request Lists

You can create a request list to store a list of requests that meet specific criteria, for example all requests that include a particular keyword in their titles, or that share a common status.

To create a request list:

1 In the Requests view, right-click the **Request Lists** node and select **Find Requests**.

2 In the Select Request dialog, enter search criteria such as title keywords, type, status, originator, and created and updated date ranges. Click Next to display all requests that meet your criteria.

3 On the new screen, select the requests you wish to include in the request list and click **Finish**.

Choosing Requests to Work On

To work on a request, you associate file revisions to it. This adds information about affected files to the requests, and maintains a record of all changes made in response to the request.

To work on a request:

1 Display your requests and find the request.

2 Right-click the request and select **Add to working list**.

3 Now, when you check out or check in files, you can relate this request to the files.

Setting an Active Request for the Dimensions CM Context View

From the Dimensions CM Explorer, you can select a request and set it as the default for the Dimensions Context view. In this way, you can spare yourself the effort of manually browsing for a specific request in the Dimensions CM Context view. When you set an

active request from the Dimensions CM Explorer view, that request is then supplied to the Context view where you can set it as the working request.

To set an active request for the Dimensions CM Context view:

1 Display your requests in the Dimensions CM Explorer.

2 Right-click a request and select **Set Active Request**. This request will now appear by default as the working request when you display the Dimensions CM Context view.

Creating a New Request

Use the following procedure to create a new request. You can also base, or prime, a new request on an existing request. If you prime an existing request, the fields in the New Request Wizard are populated with information from that request. When you create a request, it is added to the repository and assigned to the initial lifecycle state. You can leave a request in your draft (held) list if you do not have all the information when you create it.

To create a new request:

1 Do one of the following:

- In Dimensions CM Explorer, right-click Requests or Draft Requests, and choose New | Request.
- Right-click a node in the **Request List**, and choose New | Request.
- To prime with information (related project and design part) from an existing request in any list, right-click the existing request and choose New | Request.

The New Request Wizard appears.

2 From the **Raise request against** list, select the product for which the new request is to be created.

3 From the of **Type** list, select the type of request to be created.

4 For **Title**, enter a title for the request.

5 For **Description**, do one of the following:

- Select the **From file** option, and enter the name of the file containing the description of the request, or click the Browse button to find the file.
- Select the **Text** option, and enter a description of the request.

Click **Next**.

6 To base the new request on an existing request, select the **Base request on** check box and select a request from the **Base request on** list. To set a relationship between the new request and the request on which it is based, choose one from the **Relate as** list.

7 Click **Next**.

8 From the **Role Section** list, select a role. The set of attributes that are relevant to a user with that role are displayed. To set attributes, enter or select values. Attributes in italics cannot be changed. Attributes in bold are mandatory.

9 Click **Next**.

10 From the **Relate to parts** list, select the design part specifications that the request affects. If you chose to base the request on another request, parts related to that request are already populated here. To select multiple parts, use the CTRL key. To deselect a part, you can CTRL-click on the item.

11 From the **Relate to projects** list, select the project or stream that the request affects. To select multiple parts, use the CTRL key. To deselect a part, CTRL-click on the item.

12 Click **Next**.

13 To add an attachment, click **Add**, or double-click the first empty cell in the Filename column. Enter the full path to the file that you want to attach.

Double-click the **Description** field and enter a description for the attachment. If you do not add a description, Dimensions CM adds the following default text: "Attachment created <date> by <user ID>"

14 Click **Next**.

15 Review the summary of the operations that will be performed when you create the request. To save a draft of the request, select the **Save to my draft requests list** check box.

16 Click **Finish**.

Actioning a Dimensions CM Request

You can action a Dimensions CM request to its next lifecycle state, or any other valid state in the request's lifecycle. Normal Dimensions CM rules only allow transitions that are valid. After the next state has been selected, you can choose to delegate the request to a particular role.

Only users with the appropriate role for a given state can action to another state. You can action a request from your inbox or a request list.

You can action multiple requests at the same time, however they must all be the same request type, for example, CR.

To action a single request:

1 Right-click the request, and select Action. The Action Wizard opens.

2 On the Select lifecycle state page, do one of the following:

- To action to the next normal lifecycle state, select **Next lifecycle state**. If the next normal lifecycle state has more than one transition, select one from the list.
- To action to any valid state, select **To specific state**, and select a state from the list.

Click **Next**.

3 On the Modify request attributes page, do the following:

a For Role Section, select an appropriate role.

b In the Attributes table, type or select values. Attributes in bold are required and are marked with a  when they need a value. Attributes in italics cannot be modified.

Click **Next**.

4 On the Delegate to other users page, do the following:

a From the Roles list, select the role to which you want to delegate.

b For Users, select the level of responsibility for the role:

- To assign a role with sole responsibility for the request, select Leader.
- To assign a role with primary responsibility for the request, select Primary.
- To assign a role that acts as a backup to the primary role, select Secondary.

c To delegate the selected role to a user, select a user in the **Available users** list, and click Assign. You can also select multiple users.

d To replace all the users in the Assigned users list with the selected user in the Available users list, click Replace.

e To remove the role assignment from a user, select the user in the Assigned users list, and click Remove.

f To delegate all related items to these users, select **Delegate related items**.

g Click **Next**.

5 On the Add a comment page, optionally add a comment. Click **Next**.

6 On the Summary page, verify the action you are to perform on the request.

7 Click **Finish**.

To action multiple requests:

1 Select multiple requests, right-click, and select Action. The Action Requests Wizard opens.

2 On the Select lifecycle state page, from the **To specific state** list, select a state. Click **Next**.

3 On the Delegate to other users page, do the following:

a From the **Roles** list, select the role that you want to delegate.

b For Users, select the level of responsibility for the role:

- To assign a role with sole responsibility for the request, select Leader.
- To assign a role with primary responsibility for the request, select Primary.
- To assign a role that acts as a backup to the primary role, select Secondary.

c To delegate the selected role to a user, in the **Available users** list, select a user, and click **Assign**. You can also select multiple users.

d To replace all the users in the **Assigned users** list with the selected users in the **Available users** list, click **Replace**.

e To remove the role assignment from a user, select the users in the **Assigned users** list, and click **Remove**.

f To delegate all related items to these users, select **Delegate related items**.

g Click **Next**.

4 On the Add a comment page, optionally add a comment. Click **Next**.

5 On the Summary page, verify the action you are to perform on the request.

6 Click **Finish**.

Relating Objects to a Request

You can relate and unrelate different objects, such as items, requests, and projects, that are related to a request from the Eclipse interface.

To manage the objects related to a request:

1 Display the Relationships tab in the request editor.

To display the request editor, double-click the request under your Inbox or a Request List.

2 Choose from the following actions:

- To remove a related object, select it and click Delete.
- To relate a new object, select the object to relate and click Add. A dialog appears to help you select the object. For example, if you select to relate a request, the Find dialog appears to search for the request.

3 Press **Ctrl-S** to save the changes.

Note that if you close the request editor (or exit Eclipse) without saving, Eclipse will prompt you if you want to save the changes. Choose the requests to save by selecting the check boxes.

Actioning an Item Related to a Request

You can action an item related to a particular request to its next lifecycle state, or any other valid state in the request's lifecycle. Normal Dimensions CM rules only allow transitions that are valid. After the next state has been selected, you can choose to delegate the request to a particular role.

Only users with the appropriate role for a given state can action to another state. You can action a request from your request list or your Inbox.

To action an item related to a request:

1 Display the Related tab in a request editor.

To display a request editor, double-click the request.

2 Right-click on the item to action and choose from the following actions:

- **Compare**, to compare the differences between the two previous actions. You have to select two item revisions to enable compare. Compare launches the Compare Editor.
- **Action**, to action the item to another lifecycle state. This command launches the action wizard.
- **Action all In Response To**, to action all of the related items that are marked as In Response To this request. The command launches the action wizard.

- **Show History**, to display the Item Revision History view for the item.
- **Browse**, to browse the item using the editor.
- **Open Item Properties**, to display the properties for the item.

3 Press **Ctrl-S** to save the changes.

Note that if you close the request editor (or exit Eclipse) without saving, Eclipse will prompt you if you want to save the changes. Choose the requests to save by selecting the check boxes.

Delegating a Request

Delegate a request to assign it to another user. When you delegate a request, you change its role assignments. You can delegate a request from your inbox or a request list. If you are delegating multiple requests at the same time, they must be of the same type and belong to the same product catalog.

To delegate a request:

1 Right-click the request, and select Delegate.

2 From the **Select the role to delegate** list, select the role that you want to delegate.

3 For Users, select the level of responsibility for the role:

- To assign a role with sole responsibility for the request, select **Leader**.
- To assign a role with primary responsibility for the request, select **Primary**.
- To assign a role that acts as a backup to the primary role, select **Secondary**.
- To delegate the selected role to a user, in the Available users list, select a user, and click Assign.

4 To replace all of the users in the Assigned users list with the selected user in the **Available users** list, click Replace.

5 To remove the role assignment from a user, select the user in the **Assigned users** list, and click **Remove**.

6 To delegate all related items to these users, select **Delegate related items**.

7 Click **Close**.

Editing an Action Description for a Request

Use the following procedure to edit an action description for a request. You can edit a description for a request located in your Inbox or a Request List.

NOTE
You can only edit the request's detailed description from Eclipse if allowed by the process model. On the **Details** tab, the **Description** field contains the detailed description.

To edit an action description:

1 Double-click the request.

2 Click the **Comments** tab.

3 Edit the description as required.

4 Press **Ctrl-S** to save the changes.

Note that if you close the request editor (or exit Eclipse) without saving, Eclipse will prompt you if you want to save the changes. Choose the requests to save by selecting the check boxes.

The action description appears in Dimensions CM based on your defined template for the action description. This means that you can enter only text in Eclipse, and you will see html formatting when it appears in Dimensions CM. See the *Process Configuration Guide* for information on changing the action description template.

Editing Request Attributes

Use the following procedure to edit a request's attributes. You can edit attributes for a request located in your Inbox or a Request List. If you attempt to edit the attributes of multiple requests at the same time, each request will open in an individual request editor.

To edit a request's attributes:

1 Right-click a request, and select Edit Attributes.

2 From the **Role Section** list, select a role whose attributes you want to edit.

3 In the Attributes table, enter or select values. Attributes in bold are required, and attributes in italics cannot be modified.

4 Press **Ctrl-S** to save the changes.

Note that if you close the request editor (or exit Eclipse) without saving, Eclipse will prompt you if you want to save the changes. Choose the requests to save by selecting the check boxes.

If you are changing the value of an attribute that has been set to 'Is Sensitive' in the Administration Console, the Signature Authorisation dialog box appears. Enter the password for the user you are currently logged in as, and click OK. If you fail to authenticate the user successfully three times, the connection to the Dimensions CM server closes, and you have to log back in again.

Managing Request Attachments

Use the following procedure to manage the attachments to a request. You can browse the contents of the attachments, save copies to your local working area, and add or delete attachments to your request.

To manage a request's attachments:

1. Display the Attachments tab in the request editor.

   To display a request editor, double-click the request under your Inbox or a Request List.

2. Choose from the following actions:

   - To change the description of an attachment, double-click its Description field and type the new description.
   - To browse an attached file, select its row and click Browse. The file opens in the default application for the file type. The application is defined on your operating system.
   - To remove an attachment, select its row and click Delete.
   - To add a new attachment, click Add and then complete the New Attachment dialog. **File Name** is the full path and file name of the file to attach. **Attach As** is the name that you want the file to be stored as in Dimensions CM.
   - To save a copy of an attachment to your local working area, select its row, and click Save. Navigate to a folder, and click Save in the Save As dialog box.

3. Press **Ctrl-S** to save the changes.

   Note that if you close the request editor (or exit Eclipse) without saving, Eclipse will prompt you if you want to save the changes. Choose the requests to save by selecting the check boxes.

Updating Work Areas from Requests

You can use Dimensions requests to update your work area. When you do this, you copy the versions of files associated with the request to your work area. This optionally overwrite your local copies with the files from the repository.

To update from a request:

1. From the Requests view, right-click the request you want to update from and select **Update from Request**.

2. The Synchronize view appears. From here you can review the updates to your local work area as a result of updating from the request. You can see new files, changed files, deleted files, and refactored files. To confirm the update of specific files, select the files (or folders containing the files), then right-click and select **Update**. The files are updated.

If you review the project content, you will see that the local work area has been updated.