SERENA®

DIMENSIONS® CM 14.3.3

Deployment Guide

Serena Proprietary and Confidential Information

Publication date: May 2017

Table of Contents

Part 1

Serena Deployment Automation

Part 1 contains the following chapters:

Chapter 1

Introduction to Serena Deployment Automation

About Serena Deployment Automation

NOTE The default CM deployment model is Dimensions Deployment, for details see "Introduction to Dimensions Deployment" on page 33.

Serena Deployment Automation (SDA) automates software deployment, the process of moving software through pre-production stages to final production. Typically, each stage represents a step of higher criticality, such as testing to production.

Software deployment complexity increases with more releases to deploy, more deployment targets, more types of deployment targets, shortened deployment cycles, and changes in technology. Deployment Automation helps you meet the deployment challenge by providing tools that improve deployment speeds while simultaneously improving their reliability.

For full details about configuring and using SDA see the Serena Documentation Centre.

Using Dimensions CM with Deployment Automation

You can invoke an SDA application process during promotion instead of using *Dimensions Deployment*, the default CM deployment model. SDA enables you to:

- Graphically model deployment process rather than write scripts.
- Use the plug-ins to automate common deployment tasks.
- Run complex deployment processes on different machines, for example:
  - Stop a web server on machine A
  - Run an SQL script on machine B
  - If the SQL fails, run another SQL script on machine B
  - If the script succeeds, restart the web server on machine A
- Use multiple CM servers and base databases with one SDA server.
- Re-use existing SDA processes.
- Automatically promote through an SDA pipeline.
- Automate different component versions together.
- Use single sign on between CM and SDA.

NOTE The CM integration with SDA currently only supports baseline promotion and demotion.

Deployment Automation Terminology

Component

A *component* maps to a functional part of a real world application and represents a deployable artifact, such as CM baseline items. Each component can have a version.

Application

An *application* brings together all the components that need to be deployed together. They define the versions of each component and the environment the components must go through on the way to production. They also map resources, which are the hosts and machines a component needs within each environment. SDA applications are mapped to CM products.

Component process

A *component processes* defines how a component should be deployed, installed, or interacted with. A process typically contains a number of steps that are inter-dependent, and may include complex logic if a step in the process fails.

Application process

An *application process* is a high-level automation orchestration that can execute multiple component processes such as the automation of deployments and rollbacks. Application processes are executed as part of Dimensions CM baseline promotion and demotion.

Environment

An *environment* represents logical deployment locations. Your deployment processes must run against at least one environment. Environments and their resources are used by applications and components at runtime. An environment brings together components with the agent that deploys them. Deployment Automation maintains an inventory of every artifact deployed to each environment and tracks the differences between them.

Pipeline

A *pipeline* is a sequence of environments.

Agent

An *agent* is a physical resources used to run a deployment and must be installed on the target server.

Automation History

The Automation view in the web client shows details of promotions and demotions. The Queue tab lists all jobs that are waiting to be executed. The History tab lists events that have completed. You can also open SDA from the Automation view.

For details see "Viewing Automation History" on page 21.

Chapter 2

Integrating Dimensions CM and SDA

Introduction

To setup an integration between Dimensions CM and Serena Deployment Automation (SDA):

- Configure CM to use SDA, see below.
- Configure Single Sign On (SSO) between CM and SDA, see page 16.
- Map CM products to SDA applications, see page 17.
- Map stages in the CM Global Stage Lifecycle (GSL) to SDA environments, see page 18.

Configuring Dimensions CM to use SDA

To use SDA instead of Dimensions Deployment, the default CM deployment model, modify the CM configuration file.

NOTE If you do a clean install of Dimensions CM with SDA the steps described below are performed automatically by the installer.

1 Open the CM configuration file:

```
%DM_ROOT%/dm.cfg
```

2 Add these variables:

- `DM_SDA_URL`

  Specifies the URL of the SDA server, for example:

  ```
  DM_SDA_URL %DM_WEB_URL%/serena_ra
  ```

  TIP: Multiple Dimensions CM servers can use the same SDA server.

- `DM_SDA_NOTIFY_TO_USER`

  Specifies a Dimensions CM user ID that will be used by SDA to notify CM about the success or failure of automation executions. For example:

  ```
  DM_SDA_NOTIFY_TO_USER notifyusr
  ```

3 Save the file.

4 Register the notification user account password in `registry.dat` specifying `DM_SDA_NOTIFY_TO_USER` (not a real user ID) as the first parameter:

```
dmpasswd DM_SDA_NOTIFY_TO_USER -add -pwd <password>
```

5 Restart the CM Listener.

Authenticating between Dimensions CM and SDA

If you use single sign on (SSO) to authenticate between Dimensions CM and SDA, verify that the CM users who are performing configuration and deployment have privileges in SDA to list applications, components, and processes and to execute automations. SSO between CM and SDA is enabled by default if you do a clean install (not an upgrade) of both products at the same time.

If you are not using SSO, configure the integration to authenticate through the SDA service account:

1. Open the CM configuration file:

   `%DM_ROOT%/dm.cfg`

2. To specify the service account user name, which will be used to authenticate all integration actions, add this variable:

   `DM_SDA_USER <SDA user ID>`

3. Save the file.

4. Register the service account password in `registry.dat` specifying `DM_SDA_USER` (not a real user ID) as the first parameter:

   `dmpasswd DM_SDA_USER -add -pwd <sda_user_password>`

5. Restart the CM Listener.

Mapping CM Products to SDA Applications

Map your CM products to SDA applications. You can mix deployment models in the same base database; some CM products can use Dimensions Deployment and others can use SDA.

1. Log into the Dimensions CM administration console.

2. Click Product Definitions.

3. In the navigation area select a product.

4. On the Product Details tab click **Edit**.

5. For **Automation Application** select an SDA application.

   An SDA application brings together all the components that need to be deployed at the same time.

6. For **Default Process** select a default SDA application process.

   An application process defines how components should be deployed, installed, or interacted with. A process typically contains a number of steps that are inter-dependent, and may include the automation of multiple components and complex logic if a step of the process fails.

7. Click **OK**.

Mapping the Global Stage Lifecycle to SDA Environments

Map stages in the Dimensions CM Global Stage Lifecycle (GSL) to SDA environments or a pipeline. When you promote a baseline to a stage, the application process is executed as follows:

- Environment: in the mapped SDA environment
- Pipeline: in the first SDA environment in the pipeline

1 Log into the Dimensions CM administration console.

2 In Areas and Deployment click **GSL**.

3 Select the stage you want to map to SDA.

4 On the toolbar click **Properties**.

5 For **Deployment automation to** select one of the following:

- No Automation: do not map this stage to SDA (automation will not happen when promotions are made to this stage).
- Pipeline: select an SDA pipeline. A pipeline is a sequence of environments where an application process request is propagated.
- Environment: select an SDA environment against which the deployment automation process will run when promoted to the corresponding stage. An environment represents logical deployment locations.

6 Click **OK**.

Chapter 3

Using Serena Deployment Automation with CM

Introduction

When you promote a baseline using Serena Deployment Automation (SDA):

- You select the *application process* that will execute component processes and the *component version* the baseline items will be deployed to.
- The baseline items are automatically deployed into the component version, which is automatically created during the promotion and named after the baseline specification.
- The application process uses the component version to perform an automation task, for example, to deploy baseline items to the SDA environment that is mapped to the GSL stage you promoted to.

When you demote a baseline using SDA:

- The baseline is automatically demoted. The application process is executed with the earlier component version (earlier baseline) against the environment you are demoting from.
- You can demote without specifying an application process if the environment you are demoting from does not require a rollback to previous state.
- You can select a rollback process, which is specifically designed for demotion, to restore the environment you are demoting from or pick any other version of any component.

Viewing Automation History

The Automation view in the Dimensions CM web client shows details of promotions, demotions, and automations in SDA.

Viewing the Queue

The Queue tab lists all jobs that are waiting to be executed or are scheduled for execution.

Column	Description
Job ID	ID of the job. Click to display a job's properties.
Name	Name of the baseline. Click to display more information.
Detail	Baseline description.
Job State	Current state of the job. Click to display the job's status.
Queued Date	Date and time the job will execute.
Queued By	Dimensions CM user who deployed the object.
Scheduled Date	Date a job is scheduled to execute.
Created By	Dimensions CM user who created the job.
Stage	GSL stage for which the job is pending.
Application	SDA application used for the automation.
Pipeline	SDA pipeline used for the automation.
Environment	SDA environment mapped to the GSL stage where the object was promoted to or demoted from.
Process	SDA application process used to execute the automation.
Component	SDA components map to CM baselines.
Version	SDA component version name.
Product	SDA applications are mapped to CM products.

Viewing History

The History tab lists events that have completed.

Column	Description
Name	Name of the object. Click to display more information.
Detail	Baseline description.
Event Type	Promotion, Demotion, or Automation.
Reason	A description of the operation.
Event Result	Submitted, Executing, Succeeded, Failed, Canceled, or Paused. Click to display more information.

Column	Description
Event Date	Date and time of the operation.
Event By	Dimensions CM user who performed the operation.
From Stage	Stage in the GSL where the object was promoted to or demoted from.
To Stage	Stage in the GSL where the object was promoted or demoted to.
Application	SDA application used for the automation.
Pipeline	SDA pipeline used for the automation.
Environment	SDA environment mapped to the GSL stage where the object was promoted to or demoted from.
Process	SDA application process used to execute the automation. Click to open the process in SDA.
Component	SDA components map to CM baselines.
Version	SDA component version name.
Product	SDA applications are mapped to CM products.

When you select a stage in the navigation pane, event types with an associated stage or area are displayed in the content pane.

Opening the Automation View from the Desktop Client

You can open the web client Automation history view from the desktop client.

1 In a content window select any baseline.

2 From the Baseline menu select History.

3 In the History dialog box select Automation History.

Opening an SDA Process from the Web Client

1 In the web client select the Automation view.

2 Select the Queue or History tab.

3 In the Process column select an SDA process.

Opening SDA from a Baseline

Web client

1 Select the Automation or Baseline view.

2 Select a baseline.

3 On the toolbar do one of the following:

- Baseline tab: select More | Open in SDA
- Automation tab: select Open in SDA

Desktop client

1 In a content window select any baseline.

2 From the Baseline menu select Open in SDA.

Promoting Baselines

PRIVILEGES

- Promote a baseline to the next stage in the GSL: BASELINE_PROMOTE_NEXTSTAGE
- Promote a baseline to any stage in the GSL: BASELINE_PROMOTE_ANYSTAGE

1 **Web client**

- On the My Current Project tab or Baselines tab, select a baseline and click **Promote**.

Desktop client

- In a content window, select a baseline, right-click and click **Promote**.

The Promote wizard opens.

2 Select the next stage in the GSL to which you want to promote the baselines.

3 Select deployment options:

- For **Run application process** select a process that will run an automatic promotion.
- For **Deploy items to the component version** select a component version the baseline items will be deployed to. If there are multiple components used in the application process, do one of the following:
  - Select a component version for each component.
  - Accept <None> to not use corresponding components in the automation.

4 Run the application process as soon as possible or specify a time.

5 Click **Finish**.

Demoting Baselines

PRIVILEGES

- Demote a baseline to the previous stage in the GSL: BASELINE_DEMOTE_NEXTSTAGE
- Demote a baseline to any stage in the GSL: BASELINE_DEMOTE_ANYSTAGE

1 **Web client**:

- On the My Current Project tab or Baselines tab, select a baseline and click **Demote**.

Desktop client:

- In a content window, select a baseline, right-click and click **Demote**.

The Demote wizard opens.

2 Select the stage in the GSL to which you want to demote the baselines.

3 Select the **application process** that will run an automatic demotion.

For **Deploy items to the component version** select a component version the baseline items will be deployed to. If there are multiple components used in the application process do one of the following:

- Select a component version for each component.
- Accept <None> to not use corresponding components in the automation.

4 Click **Finish**.

Chapter 4

Quick Start Tutorial

Introduction

This quick start tutorial shows you how to configure an integration between CM and SDA and deploy baselines. This tutorial assumes that you have installed and configured:

- A Deployment Automation server with at least one agent.
- A Dimensions CM server and clients.

Configure SDA

NOTE

- The steps below are not required if you do a clean install of Dimensions CM with SDA and the demo schema.
- The names of process and components may not match those in the demo data.

Step 1: Create an application

Create a new application to map later to a CM product.

1 Log into SDA.

2 Navigate to Management | Applications.

3 Create a new application called `QLARIUS`.

Step 2: Create an environment

Create environments to map later to stages in the Dimensions CM Global Stage Lifecycle.

1 Select **Environments**.

2 Create these environments:

- SIT
- QA
- PRE-PROD
- LIVE

Step 3: Create a pipeline

Create a new pipeline (a sequence of environments).

1 Select **Pipelines**.

2 Create a new pipeline called `SIT_SDA`.

Step 4: Add environments to the pipeline

Add environments to the pipeline.

1 Select the pipeline SIT_SDA.

2 On the Pipeline tab click Edit.

3 Add the environments SIT and QA to the Pipeline Environment.

Step 5: Add the pipeline to the application

Add the pipeline to the application.

1 Navigate to the QLARIUS application.

2 Select **Details** and click **Edit.**

3 Add the pipeline SIT_SDA.

Step 6: Add an environment to the application

Add an environment to the application.

1 In the QLARIUS application click **Environments**. The pipeline you just added is displayed.

2 Add the environment PRE-PROD to the application.

Step 7: Create a component

Create a component that will be used to deploy CM baselines.

1 In the QLARIUS application select **Components**.

2 Create and add a new component called `QL_BLN`:

- For **Source Config Type** select File System (Basic).
- For **Base Path** enter the path to the QLARIUS baselines.

Step 8: Create and design a component process

Create and design a component process to execute an installation.

1 Navigate to the application `QLARIUS`.

2 Select **Processes**.

3 Create a new component process called `CompProcessInstall`.

4 Open the process.

5 On the **Tools** tab drag Install Component onto the process designer.

6 On the **Item Properties** tab:

a Specify a name for the process: `CompInstall`

b From the **Component** list select `QL_BLN`.

c From the **Component Process** list select `AppDownloadProcess`.

d Connect the Start and Finish steps to Install Component.

e Save the work flow and close the process designer.

Step 9: Create and design an application process

Create and design an application process to automatically download and promote baseline items.

1 In the Components view of the QLARIUS application select `QL_BLN`.

2 Select **Processes**.

3 Create a new process called `AppDownloadProcess`. The **Process Type** is Deployment.

4 Select the process.

5 On the **Tools** tab expand Repositories | Artifact | Serena DA.

6 Drag **Download Artifacts** onto the process designer.

7 Connect the *Start* and *Finish* steps to Download Artifacts.

8 Save the work flow and close the process designer.

Step 10: Create a resource

Create a new resource and map it to an SDA agent.

1 Navigate to Management | Resources.

2 Create a new resource called `sdaAgent`.

3 For **Agent** select the SDA agent installed on the host machine.

Step 11: Add a resource to an environment

Add the resource to the QA environment.

1 In the Resources view select **Environments**.

2 Select the QA environment (click the 'QA' link).

3 Select the Resources tab.

4 Add the resource `sdaAgent`.

Step 12: Add a resource to a component

Map a specific component in a specific environment to a resource. For example, map the component `QL_BLN` in the QA environment to the resource `sdaAgent`.

1 Open the QLARIUS application.

2 Select **Environments**.

3 Select the QA environment (click the 'QA' link).

4 From the pop-up menu select **View Details**.

5 Select the **Component Mapping** tab. The component `QL_BLN` should already be mapped.

6 Add the resource `sdaAgent`.

Configure Dimensions CM

Step 1: Check the CM configuration

1 Check that the Dimensions CM server configuration file (`%DM_ROOT%/dm.cfg`) has the variable `DM_SDA_URL` and points to the URL of the SDA server. In this example, SDA is installed on the same machine as the CM server:

```
DM_SDA_URL %DM_WEB_URL%/serena_ra
```

2 If you are using SSO, check that users who are configuring and promoting assets have the required privileges in SDA.

If you are not using SSO, configure the integration to authenticate through the SDA service account, see "Authenticating between Dimensions CM and SDA" on page 17.

Step 2: Map a CM product to an SDA application

Map the CM product QLARIUS to the SDA application QLARIUS.

1 In the Dimensions CM administration console select **Product Definitions**.

2 In the main window select QLARIUS.

3 For **Automation Application** select `QLARIUS`.

4 For **Default Process** select `CompProcessInstall`.

5 Click **OK**.

Step 3: Map GSL stages to SDA environments

1 In the Dimensions CM administration console, in Areas and Deployment, select **GSL**.

2 Select the SIT stage.

3 On the toolbar click **Properties**.

4 For **Deployment automation to** select Pipeline and select SIT_SDA.

5 Map the other stages to the same SDA environments.

6 Click **OK**.

Promote Baselines

1. Web client: On the My Current Project tab or Baselines tab, select a baseline and click **Promote**.

   Desktop client: In a content window, select a baseline, right-click and click **Promote**.

   The Promote wizard opens.

2. Select the next stage in the GSL to which you want to promote the baseline.

3. Select deployment options.

   - Run a process that will use a component version to perform an automation task. For **Run application process** select `CompProcessInstall`.
   - Deploy the baseline items to a component version (named after the baseline specification). For **Deploy items to the component version** select `QL_BLN`.

4. Run the application process as soon as possible.

5. Click **Finish**.

View Promotion and Automation History

To view the promotion and automation history of this baseline, log into the Dimensions CM web client and select the Automation view.

Part 2

Dimensions CM Deployment

Part 2 contains the following chapters:

Chapter 5

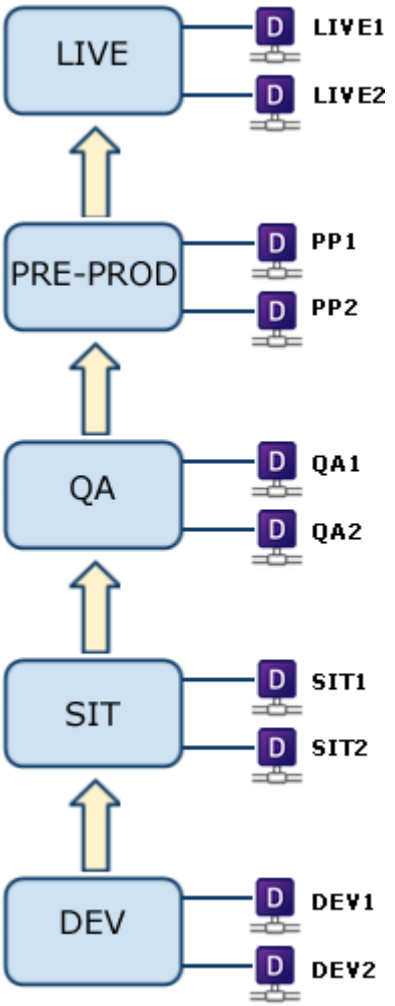
Introduction to Dimensions Deployment

Overview

NOTE Dimensions Deployment is the default CM deployment model, for details about using Serena Deployment Automation see "Introduction to Serena Deployment Automation" on page 11.

Dimensions CM manages individual items, or groups of items, as they move through the development process. The deployment change management process is defined by a series of development stages linked to the Global Stage Lifecycle (GSL). For each stage of the GSL one or more physical or logical locations on disk, known as deployment areas, can be setup to contain a snapshot of the items at that stage.

The Global Stage Lifecycle (GSL) is the base database lifecycle that items follow through the deployment process. This is a sample GSL with five stages from development (DEV) to system integration testing (SIT), quality assurance (QA), pre-production (PRE-PROD) and production (LIVE), with two deployment areas assigned to each stage:



TIP Dimensions CM executes deployment and build operations asynchronously in deployment areas. To use this functionality you should define a user name and password (or a credential set) for each deployment area.

Promoting and Demoting

To move an object from its current stage to another stage higher up the GSL you use the **Promote** command. To move an object from its current stage to another stage lower down the GSL you use the **Demote** command. For example, in the GSL illustrated above you:

- Promote from DEV to SIT.
- Demote from SIT to DEV.

TIP You can set up email notifications to alert one or more users when a promotion or demotion occurs.

Promote and Demote Rules

- You must have a privilege on the stage that you are promoting to, demoting to, or demoting from.
- Item revisions and requests in an off normal lifecycle state cannot be promoted or demoted.

By-Passing Stages in the Global Stage Lifecycle

If you have the appropriate privileges on the source and target stages you can bypass the promotion or demotion to all of the intermediate stages. For example, you might promote from DEV straight to PRE-PROD by-passing the SIT and QA intermediate stages.

NOTE Items are promoted to all intermediate stages so that they can be deployed manually to areas defined for those stages at a later time.

Deploying and Rolling Back

To put files in a deployment area you use the **Deploy** command. To remove files from a deployment area you use the **Rollback** command.

NOTE When you run the Deploy and Rollback commands, Dimensions CM fetches new copies from the repository to the target area. Files are not physically copied from one area to another, therefore, if a file is corrupted in an area it is not propagated across other areas. This also aids auditability.

TIP You can set up email notifications to alert one or more users when a deployment or rollback occurs.

Deploy and Rollback Rules

- You can only deploy files to, or roll back files from, a deployment area if the files are at the stage that owns the area or at any stage above it. For example, to deploy *foo.c* to a deployment area assigned to the SIT stage, *foo.c* must be at SIT or above. During failure recovery Dimensions may override this rule.
- The Rollback command removes an item revision from an area and automatically redeploys the item revision that was there previously. For example, if you have

revision 1.0 in an area, deploy revision 3.0, and then roll back revision 3.0, Dimensions CM redeploys revision 1.0 not revision 2.0. If no previous revision of the rolled back item was deployed to an area, the item revision is deleted. For example, if you deploy the first revision of an item and then roll it back, the revision is deleted from the area and not replaced with another revision.

- You can only roll back a complete deployment operation. An operation is all items, request and baselines that were part of the original deployment to an area. To perform a rollback you must roll back all the objects that were part of the operation. You cannot break up a group and roll back individual objects.
- If you have a privilege on a deployment area you can select or deselect it before you run the Deploy or Rollback commands.
- If a Deploy command fails it backs out everything from the failed area and stops any further deployments.
- If you deploy to multiple areas in parallel, not in sequence, and the deployment to one of these areas fails, the other deployments are not affected.
- If a Rollback command fails it puts back into the area everything that it had tried to roll back.
- If you share a deployment area with another project or stream, deploy a request (R1) to the area, and the other project then deploys a different request (R2), you cannot roll back the first request (R1) until the second request (R2) has been rolled back by the other project.
- You can only roll back in the reverse deployment order. For example, you deploy requests in the following order: R1, R2, and R3. To roll back R1 you must first roll back R3, followed by R2, and then R1.
- Item revisions, requests, and baselines in an off normal lifecycle state cannot be deployed or rolled back.

Note: Move on deploy is not supported in Dimensions CM 12.x and later.

Deployment Area Versions

All of the items in a deployment operation belong to a deployment change set. A deployment change set is then applied to a deployment area to create a new area version. An area version is therefore a delta of an area.

Using Deployment Area Filters

You can use filters to deploy specific files to a deployment area, for example, to only deploy executables to a live environment. You define and assign filters in the Area Filters section of the Dimensions administration console. For more details see the *Process Configuration Guide*.

Using Deployment Area Scripts

You can specify the following types of scripts that are executed during a deployment or rollback to an area:

- Pre deploy
- Post deploy
- Deploy failure

You specify area scripts in the Area Definitions section of the Dimensions administration console.

Difference between Promotion and Deployment

Promotion and deployment are different concepts. Promotion indicates the quality of an object and specifies what deployment areas it can be deployed to. For example, at unit test developers may be testing on their local machines and not deploying to deployment areas until system testing begins. Therefore, promoting an item to unit test means that development has been completed and unit testing is proceeding, but the item is not yet of a sufficient quality to be deployed to a deployment area.

If you do not add a deployment area to a GSL stage, promotion to that stage will not result in any deployment, but the stage still exists. You can add a deployment area later to that stage, and then deploy items to that area.

Setting a Deployment Sequence

If you are deploying to multiple areas at the same stage you can deploy in a specific order. The sequence number is initially set for a deployment area when it is assigned to a project or stream. The sequence number is honored by any deploy or rollback operation on the area. You can modify the sequence number for a deployment area in the web client.

Deployment Sequence Rules

- Areas with sequence numbers are processed first.
- The sequence numbers do not have to be contiguous.
- Areas that do not have any sequence numbers are processed last.
- Areas that share the same sequence number are grouped and deployed in an undefined sequence in their group when it is processed.
- Areas are deployed in ascending sequence starting with 0.
- When a deployment error occurs in a sequence group the other areas in that group are processed. However, any areas with higher sequence numbers are not processed.

Tip: Leave spaces between sequence numbers so that you can insert additional areas.

Deployment Sequence Example

At the LIVE stage you have the following areas and sequence numbers:

- LIVE1 (0)
- LIVE2 (5)
- LIVE3 (10)
- LIVE4 (10)
- LIVE5 (10)
- LIVE6
- LIVE7

The areas are deployed in the following sequence:

1 LIVE1

2 LIVE2

3 LIVE3, LIVE4, and LIVE5 (in no specified order)

4 LIVE6 and LIVE7 (in no specified order)

Promoting and Deploying in the same Operation

If you have the appropriate privileges you can promote and deploy in the same operation. For example, when you run the Promote command and select a stage, you can also select areas to be deployed to that are attached to the stage.

Automatic Deployment (Deploy by Default)

You can deploy files automatically (Deploy by Default) to one or more deployment areas when you run the Promote command. To use Deploy by Default:

- You must have a role to promote to the stage that the area is attached to.
- When the deployment area was setup and attached to the project or stream, the Deploy by Default option was selected.

  NOTE

  - Deploy by Default is set for each deployment area and is not a global setting.
  - When you setup Deploy by Default on an area it automatically applies to all deployment activities for that area, including rollback.

Promote with Deployment Rules

- When you select a deployment area it must be at the same stage that you are promoting to. For example, if you are promoting to the SIT stage, any area that you select must be attached to that stage. You cannot select an area that is not at the same stage.
- If the deployment area list is empty and no deployment operation has been requested, no deployment areas will be updated on the target stage during promotion.
- If you have the appropriate privilege on an area you can choose which default deployment areas of the target (promote to) stage are populated.
- If you by-pass stages during a promotion, default deployment areas assigned to the by-passed stages are not deployed to. For example, there are default deployment areas attached to the SIT and QA stages. You promote from the DEV stage straight to QA by-passing SIT. The QA default deployment areas are populated, however, the SIT default deployment areas are not deployed to.
- You do not require any privilege to deploy to default deployment areas as this privilege is implicit (permission is given automatically if you have the privilege to promote to the stage).
- You do require a privilege to deploy to non-default deployment areas as this privilege is explicit. Because the area is not a default area you must be granted the privilege on that area before you can deploy to it.

Demoting and Rolling Back in the same Operation

If you have the appropriate privileges you can roll back and demote in the same operation.

Rollback on Demotion

When you demote an object, it is automatically rolled back from default areas at higher stages. If you have privileges that allow you to demote an object, you are implicitly granted deployment privileges that allow you to perform the rollback. If the rollback cannot be performed demotion will still proceed. Be aware that rollback and demotion are independent transactions. The demotion will occur immediately, but the rollback will be queued and may not succeed.

If the object has been deployed to manual areas at a higher stage you must roll back these deployments before performing the demotion. The object will not be demoted if it is deployed to manual areas at higher stages. Rolling back from manual areas requires deployment privileges.

Manual Rollback

If you have deployment privileges for an area, you can manually roll back any deployment as long as the area version created by that deployment is not depended on by later area versions.

Viewing History

To help you decide what you need to roll back and in what order, use the History tab in the Deployment view. The tab has an option that enables you to only show objects that can be rolled back. See "The History Tab" on page 57 for details.

Demote with Rollback Rules

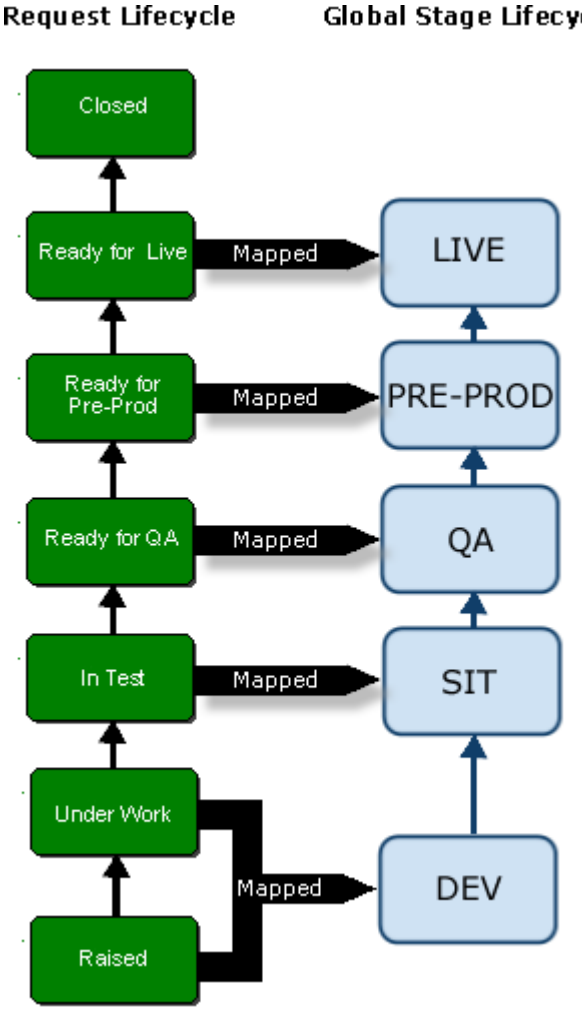
- If you have the appropriate privilege on an area you can choose which default deployment areas of the target (demote to) stage are populated.
- If the deployment area list is empty and no deployment operation has been requested, no deployment areas will be updated on the target stage during demotion.
- A demote operation will fail if any of the files are deployed to non-default areas. Under the stage being demoted from you must manually roll back those files before you can run the Demote command again.
- If a rollback from any development areas in the source stage fails, the demotion from the source stage does not complete.
- Demote automatically rolls back files in any deploy by default areas.
- When you select a deployment area it must be at the same stage that you are demoting to. For example, if you are demoting from the QA stage to SIT, any area that you select must be attached to the SIT stage. If you select an area that is not at the same stage, an error message is displayed.
- You do not require any privilege to roll back from default deployment areas as this privilege is implicit (permission is given automatically if you have the privilege to roll back from the source stage).
- You do not require any privilege to deploy to default deployment areas as this privilege is implicit (permission is given automatically if you have the privilege to deploy to the target stage).
- You do require a privilege to deploy to non-default deployment areas as this privilege is explicit. Because the area is not a default area you must be granted the privilege on that area before you can deploy to it.
- If a roll back and a demotion succeeds but the deployment then fails, a warning is displayed (only the rollback and demote operations are atomic).
- If you by-pass stages during a demotion, all default deployment areas in all by-passed stages are rolled back automatically. For example, there are default deployment areas attached to the DEV and SIT stages. You demote from the QA stage straight to DEV by-passing the SIT stage. Both the QA and the SIT default deployment areas are automatically rolled back. The Demote commands are run one at a time. If an intermediate demotion fails, any preceding successful demotions cannot be recovered. All the Demote with Rollback rules above must apply for this rule to be successful.
- A demote will fail if the objects are deployed to a deployment area where deploy by default is not enabled. You must first roll back the objects manually.
- When you rollback from a higher stage, deployment to the lower stage that you are demoting to is optional.

TIP If you have configuration files that you deploy to areas and then edit, keep them separate from your source code. For example, if you do a rollback but the rollback fails, area recovery will reinstate the previous version of the area and the configuration files will also be rolled back to their previous state. This may affect the content of the files.

Deploying Objects by Actioning

In Dimensions you can map lifecycle states to stages in the GSL. If this mapping is setup, when you action an object to a lifecycle state that is mapped to a stage, it is automatically promoted to, or demoted from, that stage. If Deploy by Default is enabled for the deployment areas attached to the stage, the object is also automatically deployed to, or rolled back from, the areas.

In this example some of the states in this request lifecycle are mapped to stages in a GSL:



For example:

- When a request is actioned up the request lifecycle from the *Under Work* state to *In Test* it is automatically promoted from DEV to SIT.
- When a request is actioned down the request lifecycle from the *Ready for QA* state to *In Test* it is automatically demoted from QA to SIT.

NOTE

- You can use action driven deployment with items, requests, and baselines.
- For action driven deployment to work you require the privileges to action and promote to stages and areas.
- Actioning an object to an off normal state removes item revisions from the associated stage.
- You map lifecycle states to the GSL in the Lifecycles section of the Dimensions CM administration console.
- When you use action driven deployment to promote or demote an item revision to another stage, the operation is applied globally to the same item revision across all projects in the current product. However, if you select the option *Use local stages* in a project/stream, the stage of the item revision in that project is not affected when the same item revision is promoted/demoted in any other project/stream. Therefore the same item revision can be at different stages in each project/stream. Once you have selected this option for a project/stream you cannot change it.

Scheduling Promotions and Deployments

When you run the Promote, Demote, Deploy, and Rollback commands you have the option to schedule the execution in the future. For example, you might want to deploy to a live production environment during the regular maintenance period when the server is offline.

NOTE If you try to deploy to an area that already has a deployment scheduled, a warning is displayed. If you proceed with your deployment the scheduled deployment may fail.

Deploying Refactoring Changes

Refactoring occurs when there are modifications in the structure of a project or stream that change the location or names of items or folders, including deletions. For example, exporting an item to a project, or moving a folder from one parent folder to another.

If you have refactoring changes you must deploy them using a request and not a baseline. If you try and deploy refactoring changes out of sequence the deployment will fail.

Always refactor using a new request that is not related to any other changes. If you do not use a new request, when you need to deploy the refactored changes (because another change is dependent on it) you might not be able to deploy the refactoring request because the other changes are related to it.

Refactoring Example

The following example describes a simple scenario where refactoring changes are deployed:

- A project contains a file, *foo.c*, at revision one, *foo.c;1*, in directory *dir1*.
- Two deployment areas, A1 and A2, are attached to the project. The stage is not relevant as the refactoring rules apply to the content of deployment areas.
- *foo.c;1* is deployed to area A1.
- foo.c is updated to create revision two, *foo.c;2*. The tip revision of foo.c in the project is now *dir1/foo.c;2*.
- foo.c is deployed to area A1, which now contains *dir1/foo.c;2*.
- foo.c is updated to create revision three, *foo.c;3*. The tip revision of foo.c in the project is now *dir1/foo.c;3*.
- If the tip revision is deployed to area A1, it will contain *dir1/foo.c;3*.
- Directory *dir1* is renamed to *dir2* using request R1. The tip revision of foo.c in the project is now *dir2/foo.c;3*.
- If *foo.c;3* is deployed to area A1 the deployment will fail because the path in the project is 'later' than the current path in the area. Request R1 needs to be deployed first as it contains the directory name change.
- foo.c is updated to create revision four, *foo.c;4*. The tip revision of foo.c in the project is now *dir2/foo.c;4*.
- *foo.c;4* cannot be deployed to area A1 until request R1 is deployed.
- However, *foo.c;4* can be deployed to area A2 because it does not contain any revisions of foo.c. Area A2 will contain *dir2/foo.c;4*.
- Deploying request R1 to area A2 will fail because A2 already contains the directory rename. *foo.c;4* must be rolled back first.

NOTE For additional refactoring examples see "Deployment and Refactoring" on page 221.

Deploying Regressions

You can use the Deploy command to overwrite an item revision with a previous revision.

- Regressions are enabled by default. For example, if you deploy an older version of an item (v3) it overwrites the latest version (v4).
- To disable regressions, set the following variable in the CM configuration file to `Yes`:

  `DM_NO_DEPLOYMENT_REGRESSION`
- To fail deployments in the event that any have been disallowed, and to display an error message, set the following variable in the CM configuration file to `Yes`:

  `DM_REGRESSION_ATTEMPT_IS_AN_ERROR`

Recommended Deployment Methods

Deploying Requests

Serena recommends that you deploy using requests for the following reasons:

- You can group all files together in a single parent request and promote and deploy the request through the GSL. Grouping ensures that all the items related to a change are progressed through the lifecycle and development areas together and no items are left behind.
- Refactoring is only fully supported by requests.

Deploying Items

Although you can deploy items there are a few disadvantages, most notably no grouping. For example, if you change multiple files, without a parent request you have to remember to always keep the files together when you move them through the GSL. This method is very error prone as it is easy to forget to promote and deploy some files.

Deploying Baselines

You can deploy baselines, however there are a number of limitations:

- A baseline that contains refactoring changes does not apply the changes to deployment areas.
- You cannot create a baseline from a deployment area, but you can build the area.
- You cannot promote, demote, deploy, or rollback non-release baselines.
- You can only promote, demote, deploy, and rollback baselines from root projects (sub-projects are not supported).
- A baseline cannot be deleted if it is currently promoted beyond the first stage in the GSL.

Auditing Deployment Areas

The audit functionality in Dimensions CM works with deployment areas and you can also use filters to ignore specific file types. You can run audits from the desktop and web clients, and the command line client. For details see the *User's Guide*, the *Process Configuration Guide,* and the *Command-Line Reference*.

Removing an Item from Deployment Areas

To remove an item from a stream or project and all associated deployment areas do the following:

1 Raise a request.

2 Prime a task from the request and delegate it to a user.

3 As that user do the following:

In a stream:

a Update the local work area.

b Delete the item from the work area.

c Deliver back to Dimensions specifying the task.

The task now has a deletion related to it.

In a project:

a Select the item.

b Do one of the following:

- *Web client*: click the More menu and select Delete.
- *Desktop client*: from the Item menu select Remove Item from Project.
- *dmcli command line client*: `RIWS "itemspec" /CHANGE_DOC=<requestN>`

c To relate the deletion to a request, in the Track changes with request field click Select and select the task that you created earlier.

d Click OK/Yes.

4 Verify that the item has been deleted from the stream or project and the DEV deployment areas.

5 Close the task.

6 Promote and deploy the request and task up the GSL, verifying at each stage that it has been deleted from the associated deployment areas.

7 Close the request.

Tip: To remove an item from a stream or project but keep it in the deployment areas, repeat steps 1 through 5 above and then close the request. Do not promote and deploy the request and task up the GSL. The item will be removed from the DEV deployment areas but will remain in all areas associated with the other stages in the GSL.

Setting the Timestamp of Deployed Items

When you browse a deployment area, the date and time that each item was last modified is displayed. You can set this timestamp as follows:

- To set the timestamp to the deployment time, define the symbol DM_TIMESTAMP_BUILDAREAS in the Dimensions configuration file, dm.cfg.
- To set the timestamp to the OS file modification time, remove the symbol DM_TIMESTAMP_BUILDAREAS from dm.cfg, or set the symbol to N.

Setting up a Deployment Environment

For information about how to set up a deployment environment see "Configuring a Deployment Environment" on page 207.

The Deployment View

The Deployment view in the web client shows details of promotions, demotions, deployments, rollbacks, builds, and various other activities related to deployment. For details see "Managing Dimensions Deployment" on page 53.

Enabling Deployment Logging

To enable deployment logging do the following:

1. Open the deployment configuration file:

   `<dm_root>\dfs\deploy_config.dat`

2. Uncomment the following line:

   `log_dir=<path>`

3. Optionally, change the directory where the logs are saved, which is specified in `<path>`.

4. Save and close the file.

5. Restart Dimensions services.

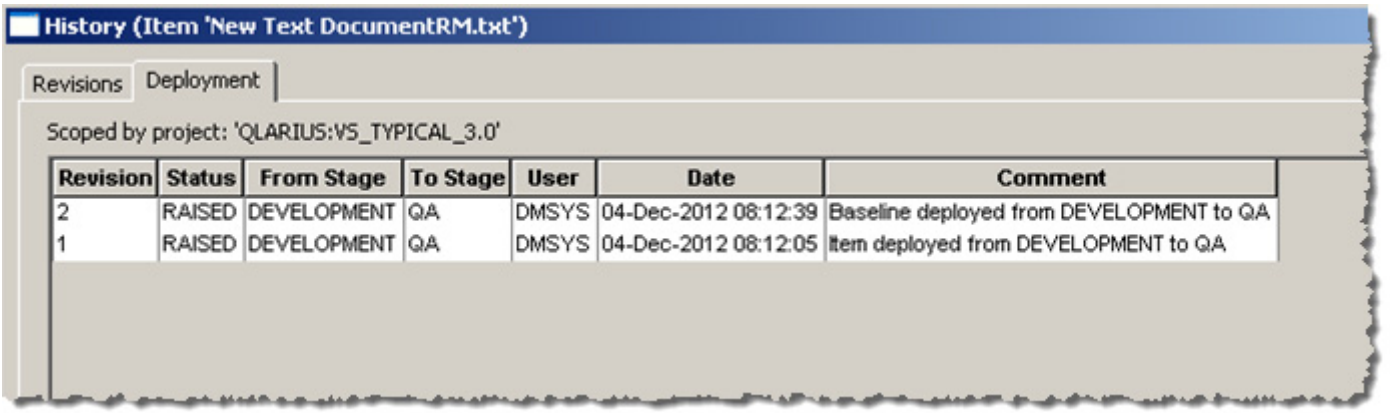
NOTE For details about configuring the deployment server, which reads the file `deploy_config.dat`, see "The Dimensions Deployment Server" on page 211.

Migrating Existing Deployment Data

You can migrate your existing deployment data from previous versions of Dimensions CM to Dimensions CM 12.x (and later) and use it with the new deployment model. For details about migration see the Windows or UNIX installation guides.

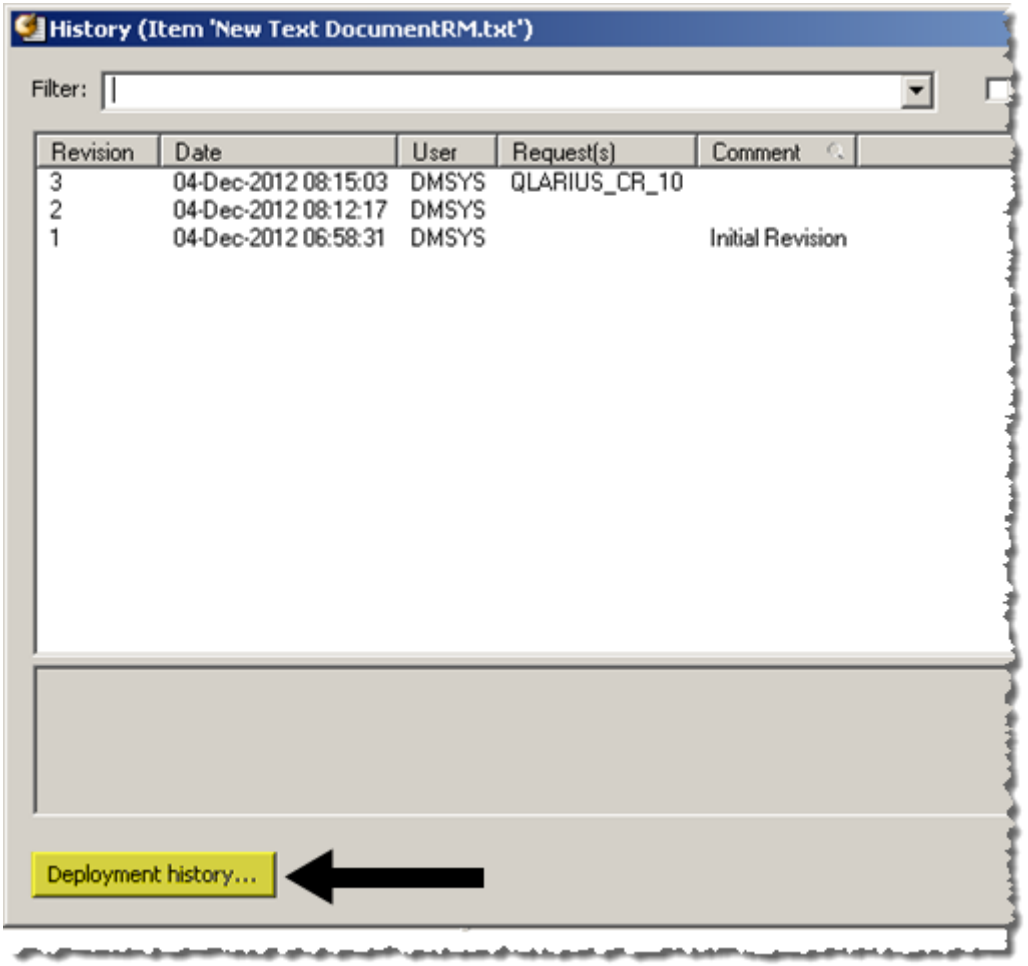
Viewing Pre-Dimensions CM 12.x Deployment History

Prior to Dimensions CM 12.x, deployment history was available in the clients. For example, when you selected history for an item, the deployment tab displayed all deployment activities:

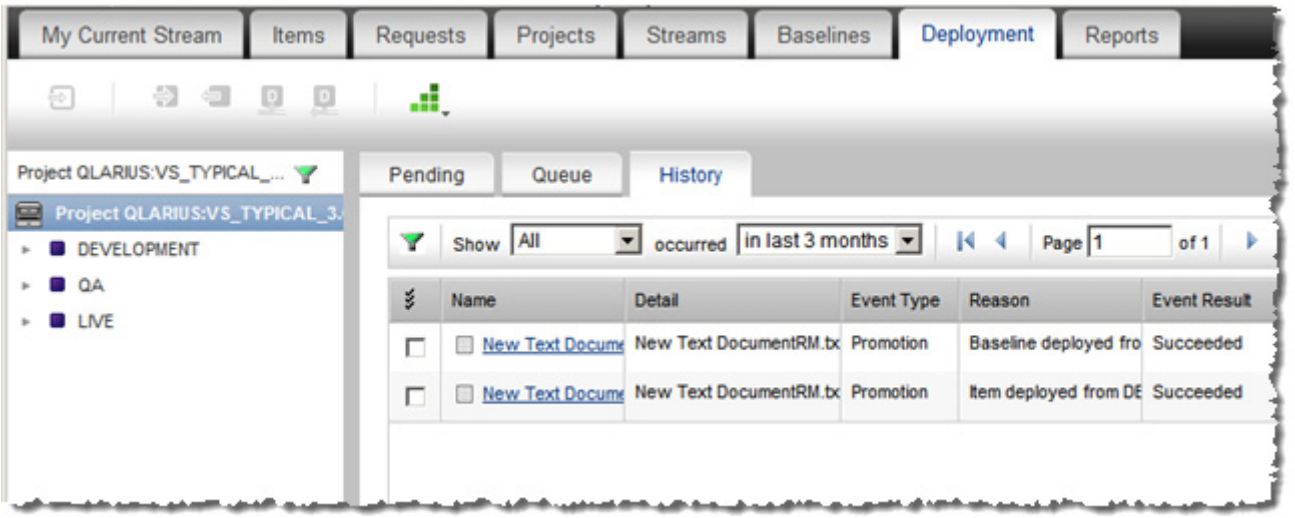


This data could also be queried through the PCMS_DEPLOYMENT_HISTORY published view.

After upgrading to Dimensions CM 12.x, all deployment data is retained. However, Dimensions CM 12.x separates the concepts of promotion and deployment. The Item history dialog box now has a Deployment History button:



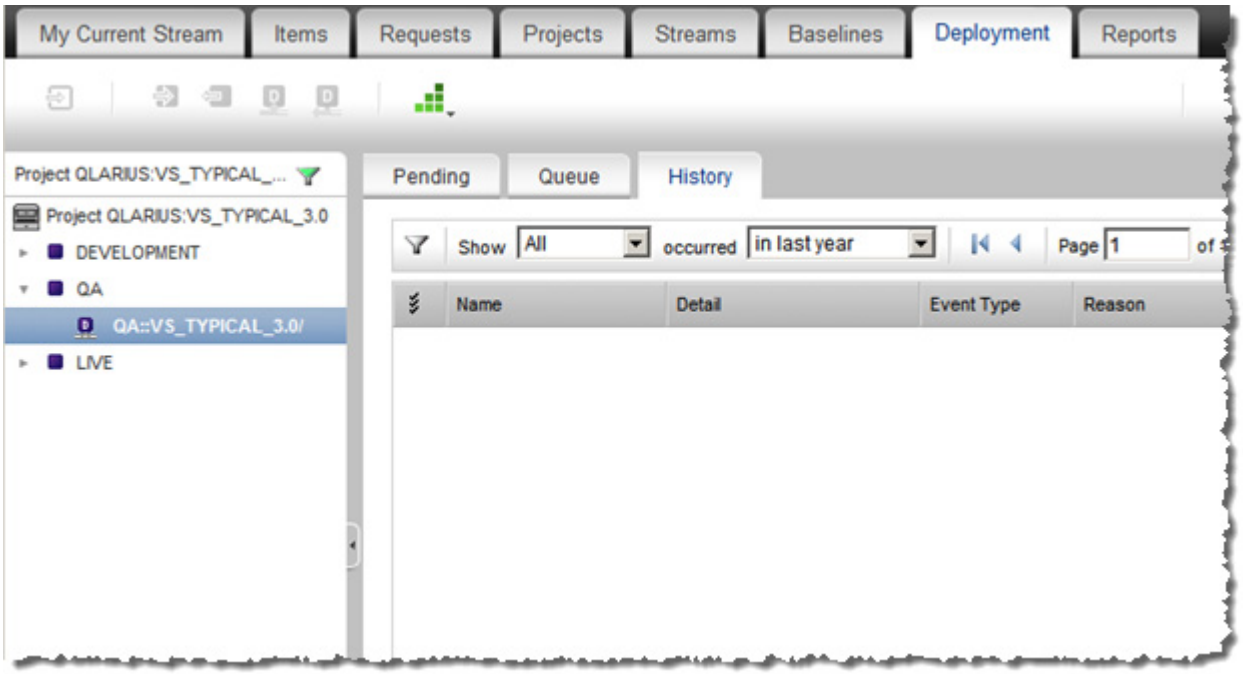
Selecting the Deployment History button takes you to the Deployment view in the web client, with the filter set for that object:



In the illustration above, all the entries are pre-Dimensions CM 12.x and their event type is 'Promotion'. History data is available at the stream or project level. There are no 'Deployment' event types.

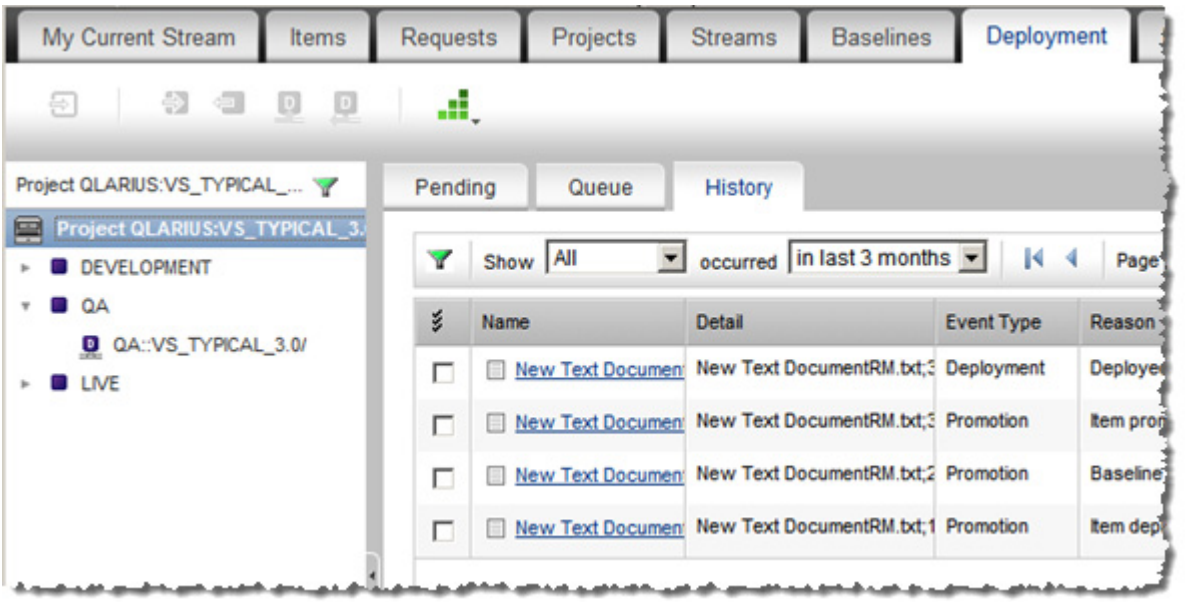
NOTE All pre-Dimensions CM 12.x data is available from the published views, however it can now only be queried from the PCMS_PROMOTE_HISTORY view.

After upgrading to Dimensions CM 12.x and migrating deployment data, you cannot browse a deployment area and view its history. Deployment areas *only* show the new 'Deployment' event type, which is not applicable to pre-Dimensions CM 12.x deployment data:

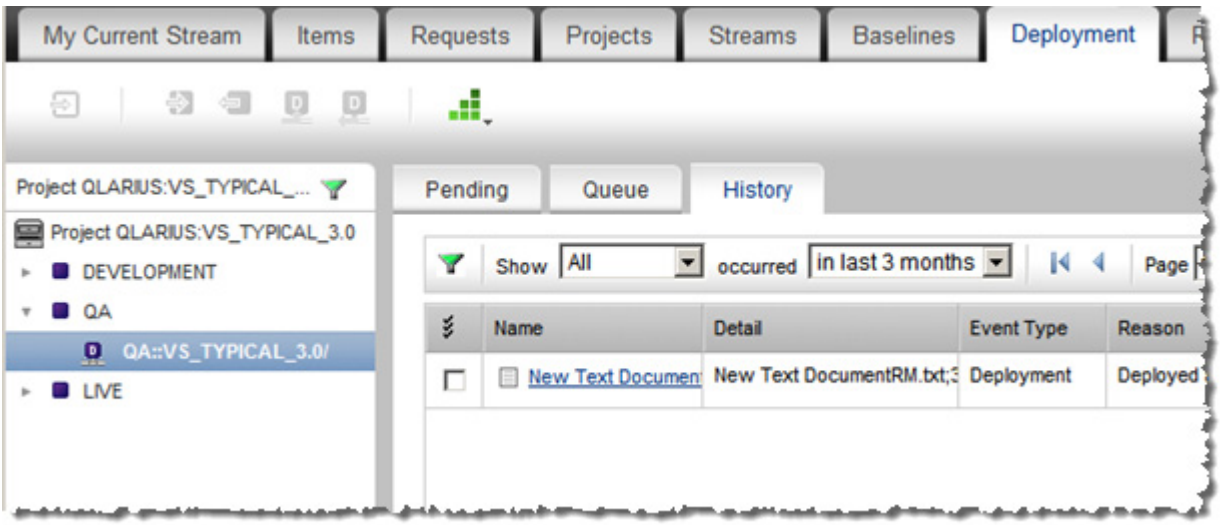


NOTE The new published view PCMS_DEPLOYMENT_DATA does not contain entries for pre-Dimensions CM 12.x deployment data.

When you promote and deploy objects in Dimensions CM 12.x you see both pre- and post Dimensions CM 12.x entries, with the 'Promotion' and 'Deployment' event types, in the history at the stream or project level:



However, if you drill down into a deployment area, *only* objects deployed in Dimensions CM 12.x, with the 'Deployment' event type, are displayed:



Deployment Commands

- PMI: promote item
- PMRQ: promote request
- PMBL: promote baseline
- DMI: demote item
- DMRQ: demote request
- DMBL: demote baseline
- SDPI: deploy item
- SDPRQ: deploy request
- SRRQ: deploy baseline
- SRAV: rollback area version

For details see the *Command-Line Reference*.

Using Library Cache Areas

When deployment areas are geographically remote from the main Dimensions server, you can associate a library cache area with a deployment area to improve deployment performance.

To associate a library cache area with a deployment area, use the CA (Create Area) or UA (Update Area) commands and supply the /LIBRARY_CACHE_AREA=<area_id> qualifier.

For example:

- The following command creates a new UAT deployment area and associates it with the ST-LC-1 library cache area:

  ```
  CA UAT1 /net=st3859 /dir=c:\deploy\uat1 /user=dmsys /pass=<password>
  /type=deployment /stage=uat /library_cache_area=st-lc-1
  ```

- The following command associates an existing UAT deployment area with the ST-LC-2 library cache area:

  ```
  UA UAT2 /library_cache_area=st-lc-2
  ```

Processing Build Jobs in Deployment Areas

When a build is performed in a deployment area, the search path sequence is:

1. The area where the build is being performed.
2. All areas specified in the build configuration that are at the same or a higher stage in the GSL as the area's stage, and which reside on the same area node.

If the area where a build is being performed is a work area, and the DM_SP_START_STAGE build template symbol has a value, the search path consists of the work area, followed by all stages at the same or higher stages as DM_SP_START_STAGE, and which reside on the same area node. In both cases, the search path is in increasing stage order, although the precise order may vary within the stage.

By default a work area is never locked and a deployment area is always locked. The BLD command qualifier [NO]LOCK_SEARCH_PATH enables you to control the locking of all areas in the search path in addition to the build area (for both deployment and work area builds).

Notes:

- For information about the DM_SP_START_STAGE build template symbol see the *Developer's Reference*.
- For information about the BLD command and the [NO]LOCK_SEARCH_PATH qualifier see the *Command Line Reference*.
- A check for whether two areas are on the same system is performed in terms of the area node, irrespective if this is a physical or a logical name. Therefore the logical name defines the system the build is being performed on.
- Any deployment operation performed for a particular user session is single threaded. Two builds from two build commands issued one after the other will execute serially. Similarly, a deployment, followed by a build, followed by a deployment to the same area, followed by another build will execute serially.

Chapter 6

Managing Dimensions Deployment

Introduction

NOTE Dimensions Deployment if the default CM deployment model. For information about using Serena Deployment Automation see "Introduction to Serena Deployment Automation" on page 11.

This chapter describes how to carry out Dimensions deployment-related functions in the web client and the desktop client. This chapter describes:

- How to use the web client Deployment view, filter what is displayed, and view details of events.
- How to assign deployment areas and view the files in deployment areas.
- How to promote, demote, deploy, and rollback items, requests, and baselines.

In the desktop client you can promote or demote items, requests, and baselines, but in order to deploy or roll back, and to manage deployment events, you need to do this from the Deployment view in the web client.

Deployment areas are defined in the Administration Console. For details, see *Area Definitions* in the online help or *Process Configuration Guide*.

You can change the GSL and add or remove stages. For details see *Editing the Global Stage Lifecycle* in the online help or *Process Configuration Guide*.

You can choose to have lifecycle states for an object type associated with stages in the GSL. This means that when items, requests, or baselines are actioned to a state associated with a deployment stage, the objects will also be promoted to that stage. If the stage has areas assigned as *Deploy by default*, the items will also automatically be deployed to those areas. This association is made using the Object type Definitions | Lifecycles application in the State Properties dialog box. For details see *How to set the State Properties for an Object Type* in the online help or *Process Configuration Guide*.

About the Deployment View

The Deployment view in the web client shows details of promotions, demotions, deployments, and rollbacks, and other activities related to deployment. The view can also be invoked from the desktop client, Visual Studio, and Eclipse.

The Deployment view displays information about:

- What was promoted, deployed, demoted, or rolled back.
- When it was promoted, deployed, demoted, or rolled back.
- Who promoted, deployed, demoted, or rolled it back.
- Where it was promoted, deployed, demoted, or rolled back.
- Was the promotion, deployment, demotion, or rollback successful.
- How it was promoted, deployed, demoted, or rolled back (by item, request, baseline, or build).

The navigation pane contains a tree structure consisting of areas grouped within stages.

The content pane contains three tabs:

- Pending
- Queue
- History

These tabs can be filtered in various ways and you can select rows and perform various operations using the toolbar buttons at the top of the Deployment view.

The Pending Tab

The Pending tab lists of all the items, requests, and baselines that have been promoted or demoted to a particular stage but not yet deployed to the areas. Objects remain in this tab until they have been deployed. The Pending tab is particularly useful for release managers as it helps them to plan and manage the deployment of software releases.

The Pending tab has the following columns:

- **Name**: the name of the object that is pending for deployment.
- **Detail**: (items only) the full path name and revision of the item.
- **Title**: (requests only) the request title.
- **Description**: (baselines only) the description of the baseline.
- **State**: the object's current lifecycle state.
- **Stage**: the GSL stage for which the object is pending.
- **Promoted Date**: the most recent date and time that the object was promoted.
- **Promoted By**: the Dimensions user who last promoted the object.
- **Area**: the deployment area where the object is currently deployed.

- **Project/Stream**: the project or stream containing the object.
- **Product**: the product that contains the project or stream.

To display more information about a request, item, or baseline, in the Name column click the object's name.

You can hide or display any of the columns and sort the list by column types.

The Queue Tab

The Queue tab lists all deployment jobs that are waiting to be executed including jobs that are:

- Scheduled for a future deployment.
- Waiting to be deployed, for example:
  - Jobs whose scheduled deployment time has passed.
  - Jobs that do not have a scheduled deployment time and are waiting to be deployed as soon as possible.

The Queue tab has the following columns:

- **Job ID**: the ID of the job.
- **Job Type**: Deployment, Rollback, Build, Clean, Audit, or Collect.
- **Job State**: the current state of the job (Submitted or Executing).
- **Queued Date**: the date and time the job will execute.
- **Queued By**: the Dimensions user who deployed the object.
- **Created By**: the user who created the job.
- **Stage**: the GSL stage for which the job is pending.
- **Area**: the deployment area where the job will be deployed.
- **Project**: the project or stream associated with the job.

To display more information about a request, item, or baseline, in the Object Name column click the object's name.

You can hide or display any of the columns and sort the list by Queued Date.

To display a job's properties, in the Job ID column click the job number. The Queued Job Properties dialog box appears and displays information about the job.

To display details about a job's status, in the Job State column click Scheduled. The Status Details dialog box appears and displays information about the job, for example:

- General messages such as 'Deployed as a result of a promotion operation'.
- Error messages.

The History Tab

The History tab lists the following events that have completed:

- Promotions
- Demotions
- Deployments
- Rollbacks
- Builds (including the collection of build outputs)
- Cleaning (of deployment areas)
- Audit (of deployment areas)

TIP

- You can use the History tab to help you plan the rollback of deployments by displaying the deployments in the order they were executed. This is the order in which they should be rolled back.
- Select **Hide if can't roll back** to hide deployments that cannot be rolled back or non-deployment events.

The History tab has the following columns (some are hidden by default):

- **Name**: the name of the object.
- **Detail**:
  - For items: the full path name and revision of the item.
  - For requests: the request title.
  - For baselines: the description of the baseline.
- **Event Type**: Deployment, Rollback, Build, Clean, Audit, Collect, Promotion, or Demotion.
- **Event Description**: a brief description of the event.
- **Reason**: a description of the transaction.
- **Event Result**: Submitted, Executing, Succeeded, Failed, Cancelled, or Paused.
- **Event Date**: the date and time of the transaction.
- **Event By**: the Dimensions user who performed the operation on the object.
- **From Stage**: the stage in the GSL where the object was promoted or demoted from.
- **To Stage:** the stage in the GSL where the object was promoted or demoted to.
- **Area**: the deployment area associated with the event.
- **Project/Stream**: the project or stream associated with the event.
- **Product** (hidden by default): the product containing the object.
- **Job ID** (hidden by default): the ID of the job to which the event belongs.

- **Revision** (hidden by default): the revision number (applies only to items).
- **Area version**: when the content of a deployment area is changed Dimensions creates a new logical version of the area. This column displays the area version that was created as a result of the event.

You can hide or display any of the columns and sort the list by Event Date.

When you select a stage node in the navigation pane, event types with an associated stage or area are displayed in the content pane. For example, deployment, rollback, build, clean, audit, collect, promotion, and demotion.

When you select an area node in the navigation pane, event types with an associated area are displayed in the content pane. For example, deployment and rollback. Promotion and demotion are not associated with an area, only a stage, and are not displayed. However, any related deployments or rollbacks may be displayed if they involve the selected area.

To display more information about a request, item, or baseline, in the Name column click the object's name.

To display details about an event's status, in the Event Result column click Failed or Succeeded. The Status Details dialog box appears and displays information about the event.

Events and Jobs

Each event is displayed on a separate row. Some Dimensions operations generate multiple events, for example, if you deploy multiple items, a separate event is generated for each item. However, all the events for this operation belong to the same job.

When you select a row in the History tab and perform an operation, the operation affects more than the selected row. For example, if you select and rollback an item, all the items that are related to that deployment job are rolled back, and not just the selected item.

Accessing the Deployment View from the Desktop Client

You can launch the web client from the desktop client by selecting **Deployment views** in various contexts. You can also do this from some of the History dialogs by clicking the Deployment History button. This displays the Deployment tab with the History tab showing related deployment/build events.

To view deployment events for the current project or stream:

Select **Deployment views** from the **Tools** menu.

To view deployment events for areas:

1. On the My Current Projects/Streams tab, in the navigation pane, expand the Deployment Areas node.
2. To view deployment events for a deployment stage, right-click the Stage node and select **Deployment views**. This will display the web client Deployment tab showing history events for that stage
3. To view deployment events for a specific deployment area, expand the Stage node, right-click the area, and select **Deployment views**.

To view deployment events for an item, request, or baseline:

In the History dialog box for one of the following:

- Items
- Request
- Baseline

Click the Deployment History button.

Using the Navigation Pane in the Deployment View

Purpose

Use the navigation pane on the Deployment tab to choose the deployment areas, projects or streams whose deployments, promotions, and demotions are displayed in the content pane.

In the navigation pane, you can select:

- All Areas
- Areas for a particular stage
- A specific area within a stage
- A specific project or stream within an area.

You can also apply a filter to the navigation pane to only show nodes for a specific project or stream.

PRIVILEGES None.

Web client

To use the navigation tree:

- To select all areas, click the top-level node.
- To select the areas for a specific Deployment stage, click a stage node.
- To select a specific area, expand the stage node and click a deployment area node.
- To select a specific project or stream within an area, click the node for the project or stream. Click the project/stream with a tick for to select your current project/stream.

To filter the navigation tree:

1. Click the filter icon at the top right of the navigation pane.
2. In the Deployment Filter dialog box:
   - To select your current project or stream, click **Show current Stream**.
   - To select another project or stream, enter the stream/project ID (`<product>:<name>`) or click the search button and use the wizard to find it. Then click the Select button.
   - To select a project or stream you have recently selected, click the name under **Recent searches...**

The name of the project or stream is displayed at the top of the navigation pane and the filter icon is now green indicating that a filter is active.

To remove the filter from the navigation tree:

1 Click the filter icon at the top right of the navigation pane.

2 In the Deployment Filter dialog box, click **Show all areas**.

Using the Lists in the Content Pane

Purpose Use the content pane to view the lists of various activities related to deployment on the Pending, Queue, and History tabs. You can page through and refresh the list

PRIVILEGES None

Web client **To page through the list**

- To display the next page, click 
- To display the previous page, click 
- To display the last page, click 
- To display the first page, click 

To refresh the deployment list

Click 

Drilling Down in the Content Pane

Purpose Drill down into rows in the content pane to view details of the jobs or the objects associated with them.

Various fields in the rows in the content pane have links that you can click on to view more details.

To view job details on the History tab

1 Click the link in the **Event Result** column for the job.

2 A Status Details window appears showing details of the event.

3 If the job is a build job and you want to view further details:

a click the job link in the Status Details window.

b Click the entry in the Build Job Details dialog box.

The Build Job Event log is displayed.

To view object details on the History, Queue, or Pending tab

To view the Open dialog for a request, item, or baseline, click the link in the **Name** column for the job.

To view job details on the Queue tab

- Click the link in the **Job ID** column.

  The Queued Job Properties window is displayed.

- Click the link in the **Job State** column.

  The Status Details window is displayed.

Changing Scheduled Jobs on the Queued Tab

Purpose

Change scheduled jobs on the Queued tab in order to reschedule the date and time they are due to start, or to cancel them.

Note that these options may not be available, for example if a job has already started executing.

PRIVILEGES Manage Schedule Jobs (ADMIN_SCHEDULING).

Web client

To reschedule a job on the Queued tab:

1. Select the job in the content pane.
2. Click the Re-schedule button.
3. In the Re-schedule Job dialog box, use the date picker to select another date and time.
4. Click OK.

To cancel a job on the Queued tab:

1. Select the job in the content pane.
2. Click the Cancel button.
3. In the Cancel Scheduled Job dialog box, click Yes.

Filtering the List in the Content Pane

Purpose Filter the lists in the content pane to reduce the entries displayed according to your chosen criteria.

PRIVILEGES None

Web client **To filter by category**

Select the category of job from the **Show** drop down list.

- For the Queued tab, you can select:
  - All
  - Queued
  - Executing
  - Scheduled
- For the Pending and History tabs, you can select:
  - All
  - Items
  - Requests
  - Baselines

The list will be updated to show your selection.

To filter jobs by queued date

Select the period from the **jobs queued** drop down list.

The list will be updated to show your selection.

To only show events that can be rolled back on the History tab

1 Click the filter icon at the top left of the History tab

2 Select **Hide if can't roll back**.

3 Click the Apply button.

To hide promotions on the Pending tab

1 Select one or more rows in the content pane.

2 Click the Hide button.

3 In the Hide Promotion dialog box, click Yes.

To include hidden promotions on the Pending tab

1 Click the filter icon at the top left of the Pending tab

2 Select **Include hidden promotions**.

3 Click the Apply button.

To show promotions on the Pending tab

1 Select Include hidden promotions on the pending tab (see above).

2 Select one or more rows in the content pane.

3 Click the Unhide button.

4 In the Unhide Promotion dialog box, click Yes.

To filter jobs queued by column values

1 Click the filter icon at the top left of the content pane.

An extra row is displayed below.

2 Click the More button.

3 Select the field you want to filter on in the left-hand drop-down list.

4 Select the comparison operator from the second drop-down list.

5 In the right-hand drop-down list, enter or select the value, or if it is a date and time use the date picker to select it.

6 If you want to add another filter criterion, click the More button. This will add another row. Repeat steps 3 through 5.

7 Click the Apply button.

To clear the filter

1 Click the filter icon at the top left of the content pane.

2 Click the Clear button.

Assigning Deployment Areas

Purpose

Assign a deployment area to a project or stream when you want to copy the item files to that area when the item revisions have reached the corresponding stage in the GSL.

You can optionally assign an area as a default area, in which case the files are automatically deployed when the item revisions are promoted to the stage.

Note that an area can be assigned a filter that determines which files are deployed to that area. This is defined in the Administration Console under Area Definitions and assigned to an area (for details, see the *Process Configuration Guide*). Each area can have only one of these filters. There is also a default Audit filter that is specified when the area is assigned to a project at a given stage. This is the default filter used when performing an Audit, but you can choose a different one (for details, see Running Builds in the User's Guide.)

NOTE When changing an area filter against an area, any previous deployments to that area which may have been filtered out cannot be redeployed from the deployment view. The recommended way to reset an area to reflect the new filter is to run an AUDIT with the FIX parameter against the area. This should add any missing files and remove any extra files based on the area filter change

PRIVILEGES
Assign Deployment Areas to Project/Stream

Web client

To assign a deployment area to a project or stream:

1. On the My Current Project/Stream tab, in the navigation pane, select the Deployment Areas node and click the Assign button.

   The Assign Area to the Current Project dialog box appears.

2. Select the area from the **Areas** list.

3. Optionally, enter the path relative to this area that you want to use as the root folder for the project in the **File path relative to area directory** field.

4. Select **Deploy by default** if you want files to automatically be deployed to this area when the item revisions are promoted or demoted to this stage.

5. Select an **Audit filter** from the list, or leave this as *Default* to use the filter assigned to the deployment area. These filters are defined in the Administration Console | Area Definitions | Filters.

   CAUTION! Audit and area filters can easily be confused. For information see *Correct Use of Area and Audit Filters* in the *Area Definitions* chapter in the *Process Configuration Guide*.

6. Enter a **Sequence order** if you want deployments to this area to occur in a particular order relative to other deployment areas when there is more than one deployment area for this project/stream. If you have no preferences, leave this as *default*. See "Setting a Deployment Sequence" on page 37 for the rules for deployment sequences.

7. If you want the appropriate item files to be copied to the area as soon as the project has been assigned, select **Populate area with project contents**.

8. Click OK.

To edit a deployment area assignment for a project or stream:

1. On the My Current Projects/Streams tab, in the navigation pane, select the Deployment Areas node.
2. Optionally, expand the Stage node for the deployment area.
3. Select the area in the content pane.
4. Click the Edit Assignment button.
5. The Edit Area Assignment dialog box appears. See "To assign a deployment area to a project or stream:" on page 64 for details of the fields to update.
6. Click OK.

To deassign a deployment area from a project or stream:

1. On the My Current Projects/Streams tab, in the navigation pane, expand the Deployment Areas node.
2. Expand the Stage node for the deployment area, and select the area.
3. Click the De-assign button.

   The De-assign Area from the Current Project dialog box appears.
4. If you want the item files to remain in the location after the assignment is removed, select **Remove files deployed from this stream**.
5. Click OK.

Desktop client

To assign a deployment area to a project/stream:

1. In the My Current Project/Stream window, right-click the Deployment Areas node and select **Assign area to project/stream**.

   The Assign an Area to the Current Project/Stream dialog box appears.
2. Select the area from the **Area** list.
3. Select a **Filter** from the list for the audit filter to be used, or leave this as *Default* to use the default filter for the area. These filters are defined in the Administration Console | Area Definitions | Filters.

   CAUTION! Audit and area filters can easily be confused. For information see *Correct Use of Area and Audit Filters* in the *Area Definitions* chapter in the *Process Configuration Guide*.
4. Optionally, enter the path relative to this area that you want to use as the root folder for the project/stream in the **Folder offset** field.
5. Select **Deploy by default** if you want files to automatically be deployed to this area when the item revisions are promoted or demoted to this stage.
6. Enter a **Sequence order** if you want deployments to this area to occur in a particular order relative to other deployment areas when there is more than one deployment area for this project/stream. If you have no preferences, leave this as *default*. See "Setting a Deployment Sequence" on page 37 for the rules for deployment sequences.
7. If you want the appropriate item files to be copied to the area as soon as the project/stream has been assigned, select **Populate area with project contents.**

8 Click OK.

To deassign a deployment area from a project/stream:

1 In the My Current Project/Stream window, expand the Deployment Areas node.

2 Expand the Stage node for the deployment area, right-click the area, and select **Deassign area from project/stream**.

3 If you want the item files to remain in the location after the assignment is removed, select **Remove files deployed from this project/stream**

4 Click Yes.

Viewing the Files in a Deployment Area

Purpose View the files in a deployment area when you want to check which files have been deployed. You can view file contents, compare file differences, and delete files in the deployment area. You can also deliver file changes in an area to the stream or project in the repository.

PRIVILEGES None

Web client **To view the contents of a deployment area for a project or stream:**

1 On the My Current Projects/Streams tab, in the navigation pane, expand the Deployment Areas node.

2 Expand the Stage node for the deployment area, and expand the node for the area you want to view. The Folders structure for the area appears beneath the area node.

3 Click a folder to view its files.

To view the contents of a file:

1 Click the name of the file in the content pane.

The Open Area File dialog box appears showing the General tab with details of the file.

2 Click the Preview tab to see the contents of the file or download a copy.

To update files from the repository (for a stream only):

Select one or more files and click the Update button.

For details of using this dialog box, see the User's Guide.

To deliver files to the repository:

Select one or more files and click the Deliver button. For see.

To compare files in the area:

Select one or more files and click the Show Differences button.

For details of using this dialog box, see the User's Guide.

To delete files in the area:

Select one or more files and click the Delete button.

MVS Areas

If you select a member in the content pane, you can perform the following operations for members that are under Dimensions CM control:

- browse
- show properties

For members that are not under Dimensions CM control, you can:

- browse
- show properties

Desktop client

To view the contents of a deployment area for a project or stream:

1. Expand the Work Areas node in the My Current Project/Stream window
2. In the My Current Project/Stream window, expand the Deployment Areas node.
3. Expand the Stage node for the deployment area, and expand the node for the area you want to view.

You can expand the folder tree on the left and display details of the item files in that folder, such as the revision and type of change (such as added or modified). If the area is located on an MVS operating system, the functions available are more limited, as described below in "MVS Areas" on page 68.

If you right-click files in this window, you can perform the following operations for files that are under Dimensions CM control:

- synchronize (for projects only)
- update
- deliver
- browse (using the editor that has been assigned in the desktop client for its file extension)
- compare
- merge (projects only)
- show item history
- show item pedigree
- show properties of the item file

For files that are not under Dimensions CM control, you can:

- browse (using the editor that has been assigned in the desktop client for its file extension)
- show properties of the file

If you right-click a folder in the tree, you can invoke the synchronize wizard to perform:

- synchronize (for projects only)
- update
- deliver.

NOTE Item files that do not belong to your current project or stream are displayed in a gray font. You can display their properties and browse them, but not perform any other operations on these files.

MVS Areas

If you right-click a member in this window, you can perform the following operations for members that are under Dimensions CM control:

- browse
- show item history
- show item pedigree

For members that are not under Dimensions CM control, you can:

- browse (using the editor that has been assigned in the desktop client for its extension)
- show properties

About Promoting

Promoting moves items upwards through the Global Stage Lifecycle (GSL).

If an associated deployment area for the project or stream has the *Deploy by Default* option set, the deployment will occur automatically upon promotion; you do not need to have the Deploy privilege. If there are other deployment areas associated with the project or stream, you can also deploy to those areas if you have the necessary privilege.

You can promote through more than one stage in the GSL in the same operation.

Note that promotion can also take place when actioning items, requests, or baselines if the next lifecycle state has been associated with a deployment stage in the GSL. If there are any default areas associated with the project/stream for that stage, the item files will also be automatically deployed to those areas.

Promoting Items

Purpose Promote item revisions in order to move them to a higher stage in the Global Stage Lifecycle, and optionally, have the item files deployed to the associated area(s).

PRIVILEGES
Promote an item to the next stage in the GSL (ITEM_PROMOTE_NEXTSTAGE).

Promote an item to any stage in the GSL (ITEM_PROMOTE_ANYSTAGE).

To deploy promoted items to a non-default area, you need the Deploy an item (ITEM_DEPLOY) privilege for the area to which you want to deploy.

Web client **To promote item revisions:**

1. On the My Current Project tab or Items tab, select one or more item revisions.
2. Click the Promote button.
3. On the first page of the Promote Wizard:
   - Select the **Next stage** to which you want to promote the item(s)
   - Optionally enter some text in the **Reason for promotion** field.
4. Click **Next**.
5. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
   - Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

     Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.
6. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
7. Click **Finish**.

Desktop client **To promote item revisions:**

1. In an Items list, select one or more items.
2. Select Item | Promote.
3. On the first page of the Promote Wizard:
   - Select the **Next stage** to which you want to promote the item(s)
   - Optionally enter some text in the **Reason for promotion** field.
4. Click **Next**.
5. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
   - Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

     Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.
6. Click **Next**. A Summary page is displayed informing you of the actions to be carried out.
7. Click **Finish**.

Promoting Requests

Purpose

Promote requests in order to move their related item revisions to a higher stage in the Global Stage Lifecycle, and optionally, have the related item files deployed to the associated area(s). Promoting a request promotes item revisions related *In response to* the request, and optionally, any child requests. This method of promotion enables you to group the item revisions that belong to a particular change and promote those items together.

Any refactoring changes that were related to the request will also be deployed.

PRIVILEGES
Promote a request to the next stage in the GSL (REQUEST_PROMOTE_NEXTSTAGE).

Promote a request to any stage in the GSL (REQUEST_PROMOTE_ANYSTAGE).

To deploy promoted requests to a non-default area, you need the Deploy a request (REQUEST_DEPLOY) privilege for the area to which you want to deploy.

Web client

To promote requests:

1. On the My Current Project tab or Requests tab, select one or more requests.
2. Click the Promote button.
3. On the first page of the Promote Wizard:
   - Select **Promote child requests** if you want items related to any child requests to also be promoted.
   - Select the **Next stage** to which you want to promote the request(s)
   - Optionally enter some text in the **Reason for promotion** field.
4. Click **Next**.
5. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
   - Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

     Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.
6. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
7. Click **Finish**.

Desktop client

To promote requests

1. Select one or more requests.
2. Select Request | Promote.
3. On the first page of the Promote Wizard:

- Select **Promote child requests** if you want items related to any child requests to also be promoted.
- Select the **Next stage** to which you want to promote the request(s)
- Optionally enter some text in the **Reason for promotion** field.

4 Click **Next**.

5 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

Promoting Baselines

Purpose Promote baselines in order to move their item revisions to a higher stage in the Global Stage Lifecycle, and optionally, have the item files deployed to the associated area(s). This method of promotion enables you to group the item revisions that belong to a particular project/stream or design part and promote them in one operation.

Note that refactoring changes will not necessarily be deployed. It is recommended that you deploy refactoring changes using requests.

PRIVILEGES
Promote a baseline to the next stage in the GSL (BASELINE_PROMOTE_NEXTSTAGE).

Promote a baseline to any stage in the GSL (BASELINE_PROMOTE_ANYSTAGE).

To deploy promoted baselines to a non-default area, you need the Deploy a baseline (BASELINE_DEPLOY) privilege for the area to which you want to deploy.

Web client **To promote baselines:**

1 On the My Current Project tab or Baselines tab, select one or more baselines.

2 Click the Promote button.

3 On the first page of the Promote Wizard:

- Select the **Next stage** to which you want to promote the baseline(s)
- Optionally enter some text in the **Reason for promotion** field.

4 Click **Next**.

5 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

Desktop client **To promote a baseline:**

1 Do one of the following:

- In a content window, select one or more baselines and select Baseline | Promote.
- In the My Current Project window, expand the sub-projects node, right-click one or more baselines, and select Promote.

2 On the first page of the Promote Wizard:

- Select the **Next stage** to which you want to promote the baseline(s)
- Optionally enter some text in the **Reason for promotion** field.

3 Click **Next**.

4 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

5 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

6 Click **Finish**.

About Demoting

Demoting moves items downwards through the Global Stage Lifecycle (GSL).

If an associated deployment area for the project or stream has the *Deploy by Default* option set, and you have selected the Perform deployment option, the deployment will occur automatically upon demotion; you do not need to have the Deploy privilege. If there are other deployment areas associated with the project or stream, you can also deploy to those areas if you have the necessary privilege.

You can demote through more than one stage in the GSL in the same operation.

Demoting Items

Purpose

Demote item revisions in order to move them to a lower stage in the Global Stage Lifecycle, and optionally, have the item files deployed to the associated area(s).

PRIVILEGES
Demote an item to the next stage in the GSL (ITEM_DEMOTE_NEXTSTAGE).

Demote an item to any stage in the GSL (ITEM_DEMOTE_ANYSTAGE).

To deploy demoted items to a non-default area, you need the Deploy an item (ITEM_DEPLOY) privilege for the area to which you want to deploy.

Web client

To demote item revisions:

1. Do one of the following:
   - **a** On the My Current Project tab or Requests tab, select one or more items.
   - **b** Click the More button and select **Demote.**

   OR

   - **a** On the Deployment tab, select the History tab, and select one or more item promote events.
   - **b** Click the Demote button.
2. On the first page of the Demote Wizard:
   - Select the **To stage** to which you want to demote the item(s)
   - If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
   - Optionally enter some text in the **Reason for demotion** field.
3. Click **Next**.
4. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.

- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

5 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

6 Click **Finish**.

Desktop client **To demote item revisions:**

1 In an Items list, select one or more items.

2 Select Item | Demote.

3 On the first page of the Demote Wizard:

- Select the **To stage** to which you want to demote the item(s)
- If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
- Optionally enter some text in the **Reason for demotion** field.

4 Click **Next**.

5 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

Demoting Requests

Purpose Demote requests in order to move them to a lower stage in the Global Stage Lifecycle, and optionally, have the item files deployed to the associated area(s). Demoting a request demotes item revisions related *In response to* the request, and optionally, any child requests.

PRIVILEGES
Demote a request to the next stage in the GSL (REQUEST_DEMOTE_NEXTSTAGE).

Demote a request to any stage in the GSL (REQUEST_DEMOTE_ANYSTAGE).

To deploy demoted requests to a non-default area, you need the Deploy a request (REQUEST_DEPLOY) privilege for the area to which you want to deploy.

Web client

To demote requests:

1. Do one of the following:
   - **a** On the My Current Project tab or Requests tab, select one or more requests.
   - **b** Click the More button and select **Demote.**

   OR

   - **a** On the Deployment tab, select the History tab, and select one or more request promote events.
   - **b** Click the Demote button.
2. On the first page of the Demote Wizard:
   - Select the **To stage** to which you want to demote the request(s)
   - If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
   - Optionally enter some text in the **Reason for demotion** field.
3. Click **Next**.
4. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
   - Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

     Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.
5. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
6. Click **Finish**.

Desktop client

To demote requests

1. Select one or more requests.
2. Select Request | Demote.
3. On the first page of the Demote Wizard:
   - Select the **To stage** to which you want to demote the request(s)
   - If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
   - Optionally enter some text in the **Reason for demotion** field.
4. Click **Next**.

5 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

Demoting Baselines

Purpose Demote baselines in order to move them to a lower stage in the Global Stage Lifecycle, and optionally, have their item files deployed to the associated area(s).

Note that refactoring changes will not necessarily be deployed. It is recommended that you deploy refactoring changes using requests.

PRIVILEGES
Demote a baseline to the next stage in the GSL (BASELINE_DEMOTE_NEXTSTAGE).

Demote a baseline to any stage in the GSL (BASELINE_DEMOTE_ANYSTAGE).

To deploy demoted baselines to a non-default area, you need the Deploy a baseline (BASELINE_DEPLOY) privilege for the area to which you want to deploy.

Web client **To demote baselines:**

1 Do one of the following:

a On the My Current Project tab or Baselines tab, select one or more baselines.

b Click the More button and select **Demote.**

OR

a On the Deployment tab, select the History tab, and select one or more baseline promote events.

b Click the Demote button.

2 On the first page of the Demote Wizard:

- Select the **To stage** to which you want to demote the baseline(s)
- If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
- Optionally enter some text in the **Reason for demotion** field.

3 Click **Next**.

4 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

5 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

6 Click **Finish**.

Desktop client **To demote a baseline:**

1 Do one of the following:

- In a content window, select one or more baselines and select Baseline | Demote.
- In the My Current Project window, expand the sub-projects node, right-click one or more baselines, and select Demote.

2 On the first page of the Demote Wizard:

- Select the **To stage** to which you want to demote the baseline(s)
- If you want the item files to be deployed to the area(s) associated with the To stage, select **Perform deployment**.
- Optionally enter some text in the **Reason for demotion** field.

3 Click **Next**.

4 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the drop-down selector to select a date. Enter or select a time in the time field.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

5 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

6 Click **Finish**.

About Deploying

Deploying results in the item files associated with the selected revisions being copied to the area(s) associated with their stage. The item revisions, must not be at a higher stage than that with which the area(s) are associated for the items to be deployed to those areas.

Deploying a request deploys the item revisions that are related *In response to* the request, and optionally, any child requests. Any refactoring changes associated with the request are also deployed.

Deploying a baseline deploys the item revisions belonging to the baseline.

Deploying can only be done from the web client.

Deploying Items

Purpose

Use this procedure to deploy one or more items to deployment areas associated with the project/stream. These items need to be at a stage that is the same or higher than the stage associated with the deployment areas.

You can also choose *either* of the following options:

- Specify a schedule (a date and time in the future when the item(s) will be deployed).
- If the related item(s) are part of a build configuration, start a build after the deployment has completed.

If you select more than one item, they must all be in the same product.

PRIVILEGES

To deploy items to a non-default area, you need the Deploy an item (ITEM_DEPLOY) privilege for the area to which you want to deploy.

You cannot deploy item revisions to an area that is associated with a stage in the GSL that is greater than the stage of those item revisions.

Web client

To deploy an item:

1. On the Deployment tab, select the Pending tab or History tab, and select one or more item entries in the list.
2. Click the Deploy button.
3. Accept the default stage, or select another stage (if available) from the **Deploy Stage** list.
4. For **Reason for deployment**, optionally type a comment.
5. Optionally, on the Deploy to Areas page:
   - If you want the deployments to be queued immediately, select **as soon as possible**.
   - If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.

- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

Deploying Requests

Purpose Use this procedure to deploy one or more requests to deployment areas associated with the project/stream. These requests need to be at a stage that is the same or higher than the stage associated with the deployment areas. Items that are related *In Response To* the request will be deployed to the selected deployment areas. Optionally, you can also deploy the items related to any child requests.

You can also choose *either* of the following options:

- Specify a schedule (a date and time in the future when the request(s) will be deployed).
- If the related item(s) are part of a build configuration, start a build after the deployment has completed.

PRIVILEGES

To deploy requests to a non-default area, you need the Deploy a request (REQUEST_DEPLOY) privilege for the area to which you want to deploy.

You cannot deploy requests to an area that is associated with a stage in the GSL that is greater than the stage of those requests.

Web client **To deploy a request:**

1 On the Deployment tab, select the Pending tab or History tab, and select one or more request entries in the list.

2 Click the Deploy button.

3 Accept the default stage, or select a stage from the **Deploy Stage** list.

4 To deploy items related to child requests, select **deploy child requests**.

5 For **Reason for deployment**, optionally type a comment.

6 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

7 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

8 Click **Finish**.

Deploying Baselines

Purpose Use this procedure to deploy one or more baselines to deployment areas associated with the project/stream. These baselines need to be at a stage that is the same or higher than the stage associated with the deployment areas.

- Specify a schedule (a date and time in the future when the baseline(s) will be deployed).
- If the deployed item(s) are part of a build configuration, start a build after the deployment has completed.

If you select more than one baseline, they must all be in the same product.

PRIVILEGES

To deploy baselines to a non-default area, you need the Deploy a baseline (BASELINE_DEPLOY) privilege for the area to which you want to deploy.

You cannot deploy baselines to an area that is associated with a stage in the GSL that is greater than the stage of those baselines.

Web client **To deploy a baseline:**

1 On the Deployment tab, select the Pending tab or History tab, and select one or more request entries in the list.

2 Click the Deploy button.

3 Accept the default stage, or select another stage (if available) from the **Deploy Stage** list.

4 For **Reason for deployment**, optionally type a comment.

5 Optionally, on the Deploy to Areas page:

- If you want the deployments to be queued immediately, select **as soon as possible**.
- If you want the deployments to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
- Under **Area(s) for deployment**, select the area(s) to which you want to deploy.

  Unless you have the necessary privilege, you will not be able to select any areas other than the default area(s) or deselect any default areas.

6 Click **Next**.

A Summary page is displayed informing you of the actions to be carried out.

7 Click **Finish**.

About Rolling Back Deployments

Rolling back a deployment removes the item revisions from an area and then automatically redeploys the item revisions that were there previously. You can only roll back a complete deployment operation consisting of all items, request and baselines that were part of the original deployment to an area.

Rolling back can only be done from the web client.

Rolling Back Items

Purpose

Use this procedure to restore the versions of files in an area to the versions that were there before an item deployment took place. Selecting one or more item revisions and selecting Roll Back shows you any versions of areas that were created as a result of deployment operations for those items, and allows you to roll back a selected version if possible.

PRIVILEGES

To roll back items from an area, you need the Rollback Item from Areas (ITEM_ROLLBACK) privilege for the project/stream/area that you want to roll back.

Web client

To roll back an item deployment:

1. On the Deployment tab, select the History tab, and select one or more item events.
2. Click the Roll Back button.
3. For **Reason for roll back**, optionally type a comment.
4. If you want the rollback to be queued immediately, select **as soon as possible**.
5. If you want the rollbacks to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
6. select the area version that you want to roll back. Area versions that cannot be rolled back have an icon.
7. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
8. Click **Finish**.

Rolling Back Requests

Purpose Use this procedure to restore the versions of files in an area to the versions that were there before a request deployment took place. Selecting one or more requests and selecting Roll Back shows you any versions of areas that were created as a result of deployment operations for those requests, and allows you to roll back a selected version if possible.

PRIVILEGES

To roll back requests from an area, you need the Rollback Item from Areas (ITEM_ROLLBACK) privilege for the project/stream/area that you want to roll back.

Web client **To roll back a request deployment:**

1. On the Deployment tab, select the History tab, and select one or more request events.
2. Click the Roll Back button.
3. Accept the default stage, or select another stage (if available) from the **Roll Back Stage** list.
4. If you want the rollback to be queued immediately, select **as soon as possible**.
5. If you want the rollbacks to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
6. select the area version that you want to roll back. Area versions that cannot be rolled back have an icon.
7. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
8. Click **Finish**.

Rolling Back Baselines

Purpose Use this procedure to restore the versions of files in an area to the versions that were there before a request deployment took place. Selecting one or more requests and selecting Roll Back shows you any versions of areas that were created as a result of deployment operations for those requests, and allows you to roll back a selected version if possible.

PRIVILEGES

To roll back baselines from an area, you need the Rollback Item from Areas (ITEM_ROLLBACK) privilege for the project/stream/area that you want to roll back.

Web client **To roll back a baseline deployment:**

1. On the Deployment tab, select the History tab, and select one or more baseline events.
2. Click the Roll Back button.

3. Accept the default stage, or select another stage (if available) from the **Roll Back Stage** list.
4. For **Reason for roll back**, optionally type a comment.
5. If you want the rollback to be queued immediately, select **as soon as possible**.
6. If you want the rollbacks to be scheduled for a particular time, select **at specified time**, and use the date picker to select a date and time.
7. select the area version that you want to roll back. Area versions that cannot be rolled back have an icon.
8. Click **Next**.

   A Summary page is displayed informing you of the actions to be carried out.
9. Click **Finish**.

Chapter 7

Deployment Scenarios

Introduction

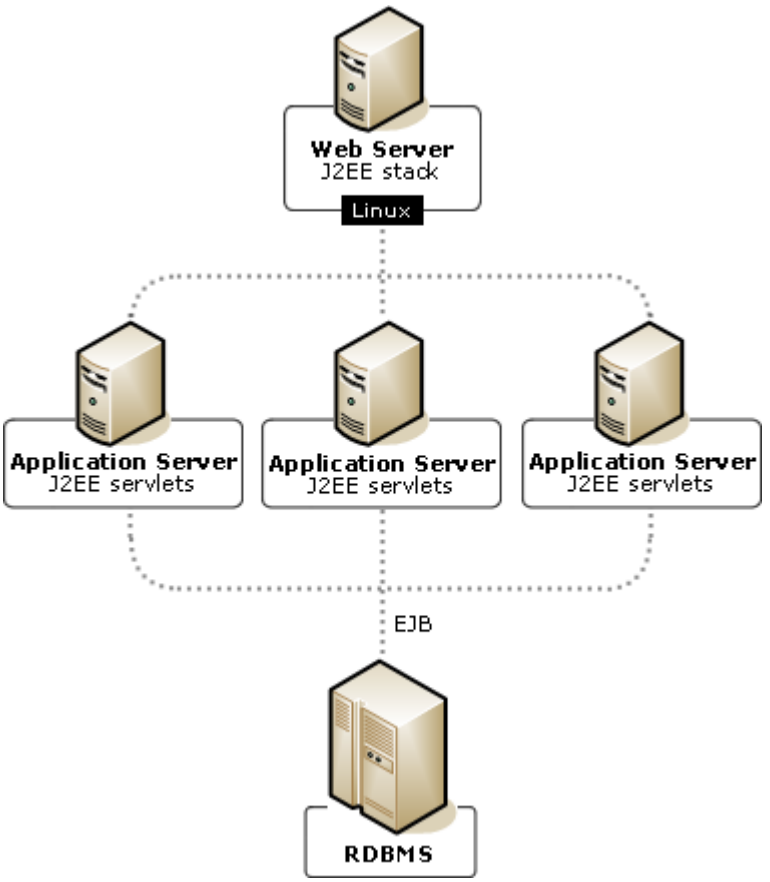
Using the Scenarios

The scenarios in this chapter describe the recommended methods for using Dimensions CM to manage deployment. You can run these scenarios in Dimensions CM by following the procedures that follow. All scenarios are performed in the web client.

About the Company

Qlarius is a web -based insurance application developed by Qlarius Health Insurance. Qlarius is built on the J2EE technology stack comprising of a browser-based user interface and a J2EE middle tier consisting of servlets. Qlarius uses EJB technology to talk to a data model that uses an RDBMS.

Development of the software takes place in London. System integration, QA, and pre-production testing are performed on a simple setup with one machine that hosts the application server and another for the RDBMS. When the application goes live it is deployed to sites in London and New York. At each of these sites there is a farm of application server machines and a separate machine hosting the RDBMS. All the sites are hosting the web application on Linux servers.



About the Company Employees

These are the company employees that you will meet in the scenarios:

- **Development Team Lead, Ted**

  Ted is the development team leader, responsible for doing design work, team leadership, tracking progress on development work, and managing the development from a technical perspective.

- **Developer, Dinesh**

  Dinesh is the Database administrator (DBA) responsible for schema design and making sure software built performs well and is making best use of the RDBMS. Dinesh is a part of Ted's team and takes technical direction from him.

- **Developer, Dawn**

  Dawn is a senior software engineer working mostly on the business logic tier of applications. Dawn is a senior member of Ted's team and often gets the most difficult and challenging technical issues to resolve.

- **Graphic Designer, Gill**

  Gill is a graphic designer who is responsible for the look and feel of Qlarius. Gill works in Ted's team as she needs to liaise closely with the developers.

- **Web developer, Wendy**

  Wendy is a web developer who is responsible for maintaining and updating the company's web sites. Wendy also works in Ted's team.

- **QA Manager, Tao**

  Tao is the QA manager responsible for making sure the applications built are fully tested and of a high quality before they are promoted to production. Tao has a team of testers who she leads, and plans and tracks their work.

- **QA Test Engineer, Tony**

  Tony is a QA engineer responsible for writing test plans, automated tests, running tests, and logging test results. Tim reports to Tao.

- **Release Manager, Rita**

  Rita is a release manager, her primary role is to ensure smooth delivery of new versions or patches to applications into their live production environment. If production goes down or there is an issue Rita is responsible for fixing it.

- **Release Build Engineer, Bobby**

  Bobby is a release build engineer responsible for managing the build automation process on-demand by running a script on the command line. Bobby saves the history of builds and releases so that he can investigate any issues that occur. Bobby is a part of Rita's team and takes technical direction from her.

- **Release Engineer, Tim**

  Tim is a release test engineer responsible for uncovering any defects, which he reports to the release manager, who then makes the decision to proceed with a release. Tim's primary role is to improve and stabilize the production and to avoid, or minimize, issues that cause defects.

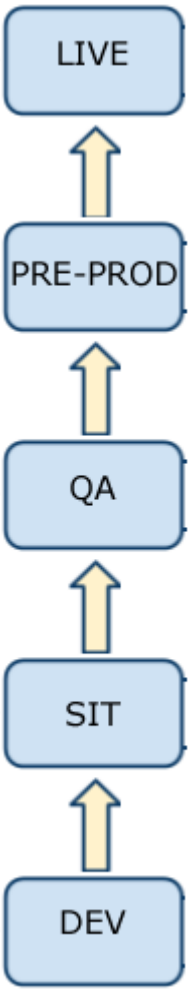
NOTE: Ask your administrator for the user IDs and passwords of these users.

About the Process Model

Qlarius Health Insurance is using the *cm_typical* process model.

The Global Stage Lifecycle

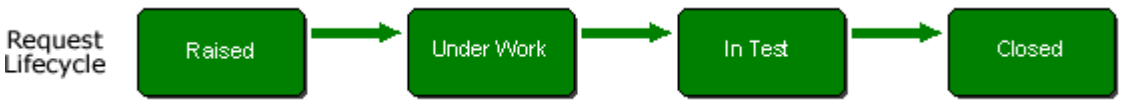
The Global Stage Lifecycle (GSL) is the base database lifecycle that items follow through the deployment process. The *cm_typical* process model has the following GSL with five stages from development (DEV) to production (LIVE):



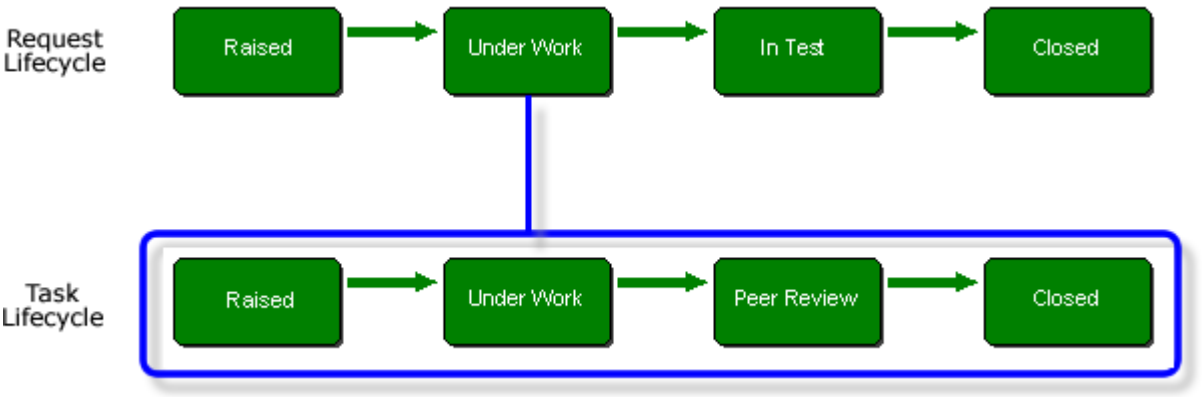
Stage	Description
DEV (Development)	Development is the initial stage where code is developed.
SIT (System Integration Test)	System Integration Test is where fixes and enhancements that have been coded, built, and tested by individual developers are brought together so that the whole system can be built and go through formal development testing.
QA (Quality Assurance)	Quality Assurance is where the QA team tests the system to ensure it is ready for the live environment.
PRE-PROD (Pre-Production)	Pre-Production is where the release team runs pre-production tests in an environment that is identical to the live environment.
LIVE	Live is where the deployment areas for the live environments are located.

The Request Lifecyles

Qlarius Health Insurance uses the following lifecycle for enhancement and defect requests:



At the Under Work state each request can be broken down into one or more child tasks, which have a different lifecycle:



About the Scenarios

This chapter contains the following scenarios:

Scenario Title	Scenario Description
Scenario 1: Basic Request Deployment	Changes are required to the corporate web site of Qlarius Health Insurance. A request is deployed to modify the web site. See page 92.
Scenario 2: Deploying Requests to a Single Deployment Area	Changes are required to the Qlarius web application. Multiple requests are deployed to a single deployment area at each stage. See page 107.
Scenario 3: Deploying Requests to Multiple Deployment Areas	Changes are required to the Qlarius web application. Multiple requests are deployed to multiple deployment areas in a specific sequence at each stage. See page 123.
Scenario 4: Deploying Refactoring Changes	Refactoring changes are deployed to the corporate web site of Qlarius Health Insurance. See page 144.
Scenario 5: Rolling Back a Deployment	There is a problem with the corporate web site of Qlarius Health Insurance. The solution is to rollback to the previous version. See page 162.
Scenario 6: Deploying a Fix Forward Solution using a Request	A defect at the QA stage is preventing testing on the Qlarius web application. The solution is to prepare a fix and deploy if forward over the part of the application that is not working. See page 180.
Scenario 7: Deploying an Emergency Fix	After an update to the corporate web site of Qlarius Health Insurance a defect is found that prevents the site from being used. The quickest solution is to apply an emergency fix. See page 192.
Scenario 8: Deploying Requests by Actioning	Changes are required to the Qlarius web application. Action driven deployment is used to deploy multiple requests to a single area at each stage. See page 203.

Pre-Requisite Steps

To run the scenarios you must perform the following pre-requisite steps:

- Install the Windows Explorer integration (in the installer this is listed as the Windows Explorer Shell Extension).
- Install the Java Development Kit on the Dimensions CM server machine (required to build the Java code).
- Install Apache Ant on the Dimensions CM server machine (required to perform builds).
- Set the build options ANT_HOME and JAVA_HOME to the location where you installed Ant and Java:

1 Log into the Dimensions CM administration console.

2 In the Distributed Development section of the Home page click Build Administration.

3 In the navigation pane expand Dimensions Projects > QLARIUS:JAVA_BRANCHA_STR.

4 Select the build configuration ANT_JAVA_BUILD and on the toolbar click Check Out.

5 Expand the Build Areas node.

6 Select the LCL_DEV_JBRNCHA_AREA03 build area.

7 In the content pane, on the Build Options tab, select the build option ANT-HOME and on the toolbar click Edit.

8 In the Value field enter the path to the Ant installation.

9 Click OK.

10 Repeat steps 7 to 9 for the build option JAVA-HOME.

11 On the toolbar click Check In.

12 Log out of the administration console.

Scenario 1: Basic Request Deployment

Scenario Objective

The objective of this scenario is to deploy requests to modify the corporate web site of Qlarius Health Insurance.

Scenario Overview

- The release manager raises an enhancement request.
- The development team lead primes a child task from the request.
- A web developer makes a modification, delivers it, and relates it to the task.
- The team lead promotes and deploys the request and task to the SIT stage and deployment area, and then promotes them to the QA stage.
- The QA manager deploys the request and task to the QA deployment area and then promotes them to the PRE-PROD stage.
- The release manager deploys the request and task to the PRE-PROD deployment area and then promotes and deploys them to the LIVE stage and production environment.

Scenario Information

- The following stream is used: QLARIUS:MAINLINE_JAVA_STR
- No build is required at any stage as only a text file is changed.
- There is a separation of duties between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- Deployment is to a single deployment area at each stage. The following deployment areas are used:

Stage	Deployment area	Deploy by Default enabled for area?
DEV	LCL_DEV_JMAIN_AREA01	Yes
SIT	LCL_SIT_JMAIN_AREA01	Yes
QA	LCL_QA_JMAIN_AREA01	No
PRE-PROD	LCL_PP_JMAIN_AREA01	No
LIVE	LCL_LIVE_JMAIN_AREA01	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 106.

Pre-Requisites

1 Create a work area on your local machine for the user Wendy, for example:

```
C:\streams\MAINLINE_JAVA_STR\wendy
```

2 Log into the web client as a user that has the privileges to promote and deploy baselines to any stage and area, for example, the tool administrator, typically *dmsys*.

3 Switch to the stream QLARIUS:MAINLINE_JAVA_STR.

4 Take a tip baseline of the stream MAINLINE_JAVA_STR.

5 To deploy the files to all deployment areas, promote and deploy the baseline as follows:

a Select the baseline and on the toolbar click Promote.

b In the Next stage field check that SIT is selected.

c Click Next.

d To deploy now, check that the option Perform deployments is set to 'as soon as possible'.

e In the Areas(s) for deployment field check that the LCL_SIT_JMAIN_AREA01 deployment area is selected. If not, select it.

f Click Next.

g A summary of the promotion and deployment activities and command that will be performed is displayed.

h Click Finish.

i Repeat step

j a' to 'h' for the other stages and their associated deployment areas:

- QA: LCL_QA_JMAIN_AREA01
- PRE-PROD: LCL_PP_JMAIN_AREA01
- LIVE: LCL_LIVE_JMAIN_AREA01

6 Log out of the web client.

Running this Scenario

Action	Procedure
The release manager raises an enhancement request	A change is required to the corporate web site of Qlarius Health Insurance. Rita, the release manager, raises an enhancement request to manage the change. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter a title for the request, for example: *Update web site* **5** In the Detailed description field enter a description, for example: *Update web site with latest corporate colors* **6** On the Attributes tab, from the Severity/Priority list select a value. **7** Click Submit and click Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.
The release manager delegates the request to the team lead	Rita delegates the request to the development team lead, Ted, whose team is responsible for maintaining the web site. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.
The release manager actions the request to its next state	Rita actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The request is removed from Rita's request inbox. **4** Log out of the web client.

Action	Procedure
The development team lead primes a child task from the request	Ted reads the email, views the request in his Request inbox, and does some design work to see what part of the web site is affected. He then primes a child task from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Request view select the Request inbox and then the request that was raised by Rita. **4** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **5** (Optional) Update the detailed description. **6** Click Submit and click Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised.
The development team lead delegates the child task to a web developer	Ted delegates the child task to a web developer, Wendy. **1** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Wendy and click Add. The wizard closes automatically. Dimensions CM sends an email to Wendy notifying her that a task has been added to her Request inbox.
The development team lead actions the task to its next state	Ted actions the task to its next state, UNDER WORK. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The child task is removed from Ted's request inbox. **4** Log out of the web client.

Action	Procedure
The web developer updates their work area from the stream	Wendy reads the email and checks her Request inbox. She does some research and identifies the file that needs to be modified, *main.css*. Wendy updates her work area from the stream. **1** Log into the web client as Wendy. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** Change Wendy's work area to the one that you created earlier (see the pre-requisites at the start of this scenario). **4** In the Items view, on the Dirs tab of the navigation pane, expand Qlarius Underwriter and select *website*. **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next. **7** Click Finish and then Close. Wendy's work area is updated.
The web developer modifies the item	In Wendy's local work area on your machine edit *main.css*. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the file.
The web developer delivers the item and relates it to the task	Wendy delivers the modification to the stream and relates it to the child task. **1** In the Items view, on the Dirs tab of the navigation pane, select *website*. **2** On the toolbar click Deliver. The Deliver to Stream wizard appears. **3** Check that the Modifications check box is selected. **4** Click Next. **5** Verify that *main.css* is selected and click Next. **6** In the Relate to request(s) field click Select. The Select Request dialog box appears. **7** From the Product name list select QLARIUS. **8** From the Type name list select TASK. **9** Click Next. **10** Select the task that is delegated to Wendy and click Finish. **11** In the Deliver to Stream wizard click Finish and then Close. **12** Make a note of the latest revision number of *main.css.*

(Sheet 3 of 12)

Action	Procedure
The developer verifies that the item was automatically deployed to the DEV area	Deploy by default is enabled for the DEV area so when Wendy delivered the item it was automatically deployed. She checks that the item was successfully deployed. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Select the History tab and in the navigation pane select the DEV stage node. **4** In the content pane verify that *main.css* was successfully deployed to the DEV deployment area LCL_DEV_JMAIN_AREA01. The Event Result column should display *Succeeded*.
The web developer delegates the task to the team lead for peer review	Wendy delegates the child task to Ted, her team lead, for peer review. **1** In the Requests view select the Request inbox. **2** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.
The web developer actions the task to its next state	Wendy actions the child task to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Wendy's Request inbox. **7** Log out of the web client.

Action	Procedure
The team lead does a peer review and actions the task to its final state	Ted has read his email, seen the task in his Request inbox, done a peer review of the file that Wendy modified, and is satisfied with the changes that she made. He actions the task to its final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox. **3** Select the child task and on the toolbar select Action. The Action wizard appears. **4** Check that the To next state field is set to CLOSED. **5** Click Finish and click OK. The task is removed from Ted's request inbox.
The team lead promotes and deploys the request and task to the SIT stage	To perform system integration testing, Ted promotes and deploys the parent request with the child task to the SIT stage and its associated deployment area. Deploy by default is enabled for the SIT area. **1** Select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that SIT is selected. **5** Click Next. **6** To deploy now, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field check that the LCL_SIT_JMAIN_AREA01 deployment area is selected. **8** Click Next. A summary of the promotion and deployment activities and command that will be performed is displayed. **9** Click Finish.

(Sheet 5 of 12)

Action	Procedure
The team lead verifies that the promotion and deployment were successful	Ted verifies that the promotion and deployment operations were executed successfully. **1** Select the Deployment view and check that only the current stream is displayed. **2** Check that the History tab is selected. **3** In the navigation pane select the SIT stage node. **4** In the content pane verify that the request was promoted successfully from DEV to SIT. The Event Result column should display *Succeeded*. **5** In the navigation pane expand the SIT stage node and select the LCL_SIT_JMAIN_AREA01 deployment area. **6** In the content pane verify that the request and task were successfully deployed to the SIT deployment area. **7** To browse the SIT deployment area, in the My Current Project view expand *Deployment Areas > SIT Stage > LCL_SIT_JMAIN_AREA01 > Qlarius Underwriter > website* **8** In the content pane verify that the latest revision of *main.css* was deployed.
Ted performs system integration testing.	
The team lead promotes the request and task to the QA stage	System integration testing has been completed successfully so Ted promotes the request and task to the QA stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the QA deployment area. **1** Select the Deployment view and in the navigation pane select the SIT stage node. **2** Select the History tab and in the content pane select the request. **3** On the toolbar click Promote. The Promote wizard appears. **4** Check that the option Promote child requests is selected. **5** In the Next stage field check that QA is selected. **6** Click Next. Deploy by Default is not enabled so no deploy options are available. **7** Click Next. A summary of the promotion activity and command that will be performed is displayed. **8** Click Finish. **9** In the navigation pane select the QA stage node. **10** In the content pane verify that the request was promoted successfully from SIT to QA.

Action	Procedure
The team lead actions the request to its next state	Ted actions the request to its next lifecycle state, IN TEST, so that the QA team can perform testing. **1** On the History tab select the request. **2** On the toolbar click Action. The Action wizard appears. **3** In the New State section check that the To next state field is set to IN TEST. **4** Click Next. **5** In the Details of solution given field enter: *main.css updated* **6** Click Finish. Dimensions CM sends an email to Tao, the QA manager, notifying her that a task has been added to her Request inbox **7** Log out of the web client.

(Sheet 7 of 12)

Action	Procedure
The QA manager deploys the request and task to the QA deployment area	Tao, the QA manager, reads the email and checks the Pending tab for the QA stage on the Deployment view. Tao sees that the request is ready to be deployed to QA. **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Deployment view check that only the current stream is displayed. **4** Select the Pending tab. **5** In the navigation pane select the QA stage node. **6** In the content pane, from the Show list select Requests. **7** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **8** Check that the option Deploy child requests is selected. **9** Check that the Deploy Stage is set to QA. **10** Click Next. **11** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **12** In the Areas(s) for deployment field check that the LCL_QA_JMAIN_AREA01 deployment area is selected. **13** Click Next. A summary of the deployment activity and command that will be performed is displayed. **14** Click Finish. **15** Select the History tab. **16** In the navigation pane expand the QA stage node and select the LCL_QA_JMAIN_AREA01 deployment area. **17** In the content pane verify that the request and child task were successfully deployed to the QA area.
The QA team performs their tests.	

(Sheet 8 of 12)

Action	Procedure
The QA manager promotes the request and task to the PRE-PROD stage	QA testing has been complete successfully so Tao promotes the request and task to the PRE-PROD stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the PRE-PROD deployment area. **1** On the History tab select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that PRE-PROD is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the navigation pane select the PRE-PROD stage node. **9** In the content pane verify that the request was promoted successfully from QA to PRE-PROD.
The QA manager actions the request to its final lifecycle state	Tao closes the request. **1** On History tab select the request. **2** On the toolbar click Action. The Action wizard appears. **3** Check that the To next state field is set to CLOSED. **4** Click Finish and click OK. **5** Log out of the web client.

Action	Procedure
The release manager deploys the request and task to the PRE-PROD deployment area	Rita, the release manager, checks the Pending tab for the PRE-PROD stage on the Deployment view. Rita sees that the request is ready to be deployed to PRE-PROD. **1** Log into the web client as Rita. **2** In the Deployment view check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane select the PRE-PROD stage node. **5** In the content pane, from the Show list select Requests. **6** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to PRE-PROD. **9** Click Next. **10** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the LCL_PP_JMAIN_AREA01 deployment area is selected. **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish and click Close. **14** Select the History tab. **15** In the navigation pane, in the PRE-PROD stage node, select the LCL_PP_JMAIN_AREA01 deployment area. **16** In the content pane verify that the request and child task were successfully deployed to the PRE-PROD area.
The release team performs their tests.	

(Sheet 10 of 12)

Action	Procedure
The release manager promotes the request and task to the LIVE stage	Rita promotes the request and task to the LIVE stage. Deploy by Default is not enabled for the LIVE deployment area. **1** On the History tab, with the PRE-PROD stage node selected in the navigation pane, select the request in the content pane. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that LIVE is selected. **5** Click Next. Rita has the privilege to deploy at the same time as the promotion but chooses not to select any deployment areas. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the navigation pane select the LIVE stage node. **9** In the content pane verify that the request was promoted successfully from PRE-PROD to LIVE.

(Sheet 11 of 12)

Action	Procedure
The release manager deploys the request to the LIVE deployment area	Let's assume that it is now the regular maintenance period when the LIVE deployment area is offline. Rita checks to see what requests are ready to be deployed to the LIVE deployment area. **1** In the Deployment view select the Pending tab. **2** In the navigation pane select the LIVE stage node. **3** In the content pane select the request and on the toolbar click Deploy. The Deploy wizard appears. **4** Check that the option Deploy child requests is selected. **5** Check that the Deploy Stage is set to LIVE. **6** Click Next. **7** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **8** In the Areas(s) for deployment field check that the LCL_LIVE_JMAIN_AREA01 deployment area is selected. **9** Click Next. A summary of the deployment activity and command that will be performed is displayed. **10** Click Finish and click Close.
The release manager verifies that the deployment operation was successful	Rita verifies that the deployment operation was successful. **1** Select the History tab. **2** In the navigation pane expand the LIVE stage node and select the LCL_LIVE_JMAIN_AREA01 area. **3** In the content pane verify that the request and task were successfully deployed to the LIVE area.
The release manager verifies that the correct item revision was deployed	Rita verifies that the correct item revision was deployed to the LIVE area. **1** In the My Current Project view, in the navigation pane, expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JMAIN_AREA01* > *Qlarius Underwriter* > *website* **2** In the content pane verify that the latest revision of `main.css` was deployed.
End of scenario	

(Sheet 12 of 12)

Scenario Privileges

The tables below list the promotion and deployment privileges required by the users in the above scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT
	QA Manager	QA
	Release Manager	PRE-PROD

Deployment privilege	Privilege owner	Required for these areas
The SIT area is a deploy by default area and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	QA Manager	LCL_QA_JMAIN_AREA01
	Release Manager	LCL_PP_JMAIN_AREA01 LCL_LIVE_JMAIN_AREA01

Scenario 2: Deploying Requests to a Single Deployment Area

Scenario Objective

The objective of this scenario is to deploy requests to modify the Qlarius web application. Deployment is to a single deployment area at each stage. This scenario is similar to "Scenario 1: Basic Request Deployment" on page 92, the main difference is a build at the DEV stage.

Scenario Overview

- The release manager raises an enhancement request.
- The development team lead primes a child task from the request.
- A developer makes a modification, delivers the modification and relates it to the task, builds the task, and captures the build outputs in Dimensions CM.
- The team lead promotes and deploys the request and task to the SIT stage and deployment area, and then promotes them to the QA stage.
- The QA manager deploys the request and task to the QA deployment area and then promotes them to the PRE-PROD stage.
- The release manager deploys the request and task to the PRE-PROD deployment area and then promotes and deploys them to the LIVE stage and production environment.

Scenario Information

- The following stream is used: QLARIUS:JAVA_BRANCHA_STR
- There is a separation of duties between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- A build is only required at the DEV stage.
- Deployment is to a single deployment area at each stage. The following deployment areas are used:

Stage	Deployment area	Deploy by Default enabled on area?
DEV	LCL_DEV_JBRNCHA_AREA03	Yes
SIT	LCL_SIT_JBRNCHA_AREA03	Yes
QA	LCL_QA_JBRNCHA_AREA03	No
PRE-PROD	LCL_PP_JBRNCHA_AREA03	No
LIVE	LCL_LIVE_JBRNCHA_AREA03	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 122.

Pre-Requisite

Create a work area on your local machine for the user Dinesh, for example:

`C:\streams\JAVA_BRANCHA_STR\dinesh`

Running this Scenario

Action	Procedure
The release manager raises an enhancement request	Changes are required to the Qlarius web application. Rita, the release manager, raises an enhancement request to manage the change. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter: *Implement support for Qlarius iPhone app* **5** In the Detailed description field enter a description. **6** On the Attributes tab, from the Severity/Priority list select a value. **7** Click Submit and then Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.
The release manager delegates the request to the team lead	Rita delegates the request to the development team lead, Ted, whose team is responsible for maintaining Qlarius. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.
The release manager actions the request to its next state	Rita actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and then OK. The request is removed from Rita's request inbox. **4** Log out of the web client.

Action	Procedure
The development team lead primes a child task from the request	Ted reads the email, views the request in his Request inbox, and does some design work to see what part of the application is affected. He then primes a child task from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view select the Request inbox and then the request that was raised by Rita. **4** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **5** (Optional) Update the detailed description. **6** Click Submit and then Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised
The development team lead delegates the task to a developer	Ted delegates the child task to a Dinesh, one of the developers in his team. **1** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Dinesh and click Add. The wizard closes automatically. Dimensions CM sends an email to Dinesh notifying him that a task has been added to his Request inbox.
The development team lead actions the task to its next state	Ted actions the task to its next state, UNDER WORK. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and then OK. The child task is removed from Ted's request inbox. **4** Log out of the web client.

(Sheet 2 of 14)

Action	Procedure
The developer updates their work area from the stream	Dinesh reads the email and checks his Request inbox. He does some research and identifies the item that needs to be modified, *LifeQuote.java.* He updates his work area from the stream. **1** Log into the web client as Dinesh. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Change Dinesh's work area to the one that you created earlier (see the pre-requisites at the start of this scenario). **4** In the Items view, on the Dirs tab of the navigation pane, expand Qlarius Underwriter > qlarius and select *interfaces*. **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next. **7** Click Finish and then Close. Dinesh's work area is updated.
The developer modifies the item	In Dinesh's local work area edit the item, *LifeQuote.java*. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the item.
The developer delivers the item and relates it to the task	Dinesh delivers the modification to the stream and relates it to the child task. **1** In the Items view, on the Dirs tab of the navigation pane, select *interfaces*. **2** On the toolbar click Deliver. The Deliver to Stream wizard appears. **3** Check that the Modifications check box is selected. **4** Click Next. **5** Verify that *LifeQuote.java* is selected and click Next. **6** In the Relate to request(s) field click Select. The Select Request dialog box appears. **7** From the Product name list select QLARIUS. **8** From the Type name list select TASK. **9** Click Next. **10** Select the task that is delegated to Dinesh and click Finish. **11** In the Deliver to Stream wizard click Finish and then Close.

(Sheet 3 of 14)

Action	Procedure
The developer verifies that the item was automatically deployed to the DEV area	Deploy by default is enabled for the DEV area so when Dinesh delivered the item it was automatically deployed. He checks that the item was successfully deployed. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Check that the History tab is selected. **4** In the navigation pane select the DEV stage node and then the LCL_DEV_JBRNCHA_AREA03 deployment area. **5** In the content pane verify that *LifeQuote.java* was successfully deployed. The Event Result column should display *Succeeded*.

(Sheet 4 of 14)

Action	Procedure
The developer builds the task and captures the outputs	Dinesh builds the task in the DEV deployment area and captures the build outputs in Dimensions CM against the task. **1** In the Request view select the child task, on the toolbar click Build, and select Build. The Run Build wizard appears. **2** In the Build Configuration field accept the default configuration. **3** From the Build Stage list select DEV. **4** From the Build Area list select LCL_DEV_JBRNCHA_AREA03. **5** Click Next. **6** Select the option Check in build outputs automatically. This check the build outputs into Dimensions CM. **7** To specify the request that the build outputs will be related to when they are checked into Dimensions CM, in the Specify the request(s) field click Select. The Selection wizard appears. **8** From the Product name list select QLARIUS. **9** From the Type name list select TSK. **10** Click Next. **11** Select the child task that is delegated to Dinesh. **12** Click Finish. The Selection wizard closes. **13** In the Run Build wizard click Next. **14** Accept the default build option selections (none) and click Next. **15** Accept the default target selection options. **16** In the target list select *Jar Files.* **17** Click Next. A summary of the build command that will be executed is displayed. **18** Click Finish. **19** (Optional) To monitor the progress of the build click the Job ID link. **20** Click Close.

(Sheet 5 of 14)

Action	Procedure
The developer verifies that the build was successful and that the outputs were captured in Dimensions CM and deployed	Dinesh verifies that the build was successful and that the outputs were captured in Dimensions CM against the task and deployed to the DEV deployment area. **1** To verify that the build was successful, in the Deployment view select the History tab. **2** In the navigation pane expand the DEV stage node and select LCL_DEV_JBRNCHA_AREA03. **3** In the content pane verify that the Event Result column displays *Succeeded* for the following objects: • Lifequote.jar (the Event Type is Collect) • QLARIUS:JAVA_BRANCHA_STR (the Event Type is Build) **4** Open *Lifequote.jar*, make a note of the item revision, and click Cancel. **5** To verify that the build outputs were captured in Dimensions CM, in the Requests view select the Request inbox and open the task. The Open Request dialog box appears. **6** Select the Relationships tab and from the Related object class list select Items. A list of all the build outputs that were captured in Dimensions CM against the task is displayed. **7** Click Cancel. **8** To browse the DEV deployment area, in the My Current Project view expand *Deployment Areas > DEV Stage > LCL_DEV_JBRNCHA_AREA03* **9** Select the folder *Qlarius Underwriter*. **10** In the content pane verify that the latest revision of *LifeQuote.jar* was deployed.
The developer delegates the task to the team lead	Dinesh delegates the task to Ted for peer review. **1** In the Requests view select the Request inbox. **2** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.

(Sheet 6 of 14)

Action	Procedure
The developer actions the task to its next state	Dinesh actions the task to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Dinesh's request inbox. **7** Log out of the web client.
The team lead does a peer review and actions the task to its final state	Ted has read his email, seen the task in his Request inbox, done a peer review of the item that Dinesh modified, and is satisfied with the changes that he made. He actions the task to its final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox, select the child task, and on the toolbar click Action. The Action wizard appears. **3** Check that the To next state field is set to CLOSED. **4** Click Finish and click OK. The child task is removed from Ted's request inbox.

Action	Procedure
The team lead promotes and deploys the parent request to the SIT stage	To perform system integration testing, Ted promotes and deploys the parent request and task to the SIT stage and its associated deployment area. Deploy by default is enabled for the SIT area. A build is not required at the SIT stage so the team lead uses an area filter to only deploy the items that are required for testing at SIT (the executables). **1** Select the parent request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that SIT is selected. **4** Click Next. **5** To deploy immediately, check that the option Perform deployments is set to 'as soon as possible'. **6** In the Areas(s) for deployment field select the LCL_SIT_JBRNCHA_AREA03 deployment area. Note: If both areas are selected, deselect SIT LCL_SIT_JBRNCHA_AREA01. **7** Click Next. A summary of the promotion and deployment activities and command that will be performed is displayed. **8** Click Finish.
The team lead verifies that promotion and deployment were successful	Ted verifies that promotion and deployment operations were successful and that the executables were deployed to the SIT area. **1** Select the Deployment view and check that only the current stream is displayed. **2** Check that the History tab is selected and in the navigation pane select the SIT stage node. **3** In the content pane verify that the request was promoted successfully from DEV to SIT. **4** In the navigation pane expand the SIT stage node and select the LCL_SIT_JMAIN_AREA03 deployment area. **5** In the content pane verify that the request and child task were deployed successfully to the SIT deployment area. **6** To browse the SIT deployment area, in the My Current Project view expand *Deployment Areas > SIT Stage > LCL_SIT_JBRNCHA_AREA03.* **7** Select the folder *Qlarius Underwriter*. **8** In the content pane verify that *LifeQuote.jar* was deployed.
Ted performs system integration testing.	

(Sheet 8 of 14)

Action	Procedure
The team lead promotes the request and task to the QA stage	System integration testing has been completed successfully so Ted promotes the request and task to the QA stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the QA deployment area. **1** In the Requests view select the Request inbox and then the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that QA is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the Deployment view select the History tab and in the navigation pane select the QA stage node. **9** In the content pane verify that the request was promoted successfully from SIT to QA.
The team lead actions the request to its next state	Ted actions the request to its next lifecycle state, IN TEST. **1** On the History tab select the parent request and on the toolbar click Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to IN TEST. **3** Click Next. **4** In the Details of solution given field enter: *Lifequote updated* **5** Click Finish. Dimensions CM sends an email to Tao, the QA manager, notifying her that a request has been added to her Request inbox **6** Log out of the web client.

(Sheet 9 of 14)

Action	Procedure
The QA manager deploys the request and task to the QA deployment area	Tao reads the email, checks her Request inbox, and deploys the request and task to the QA deployment area. A build is not required at the QA stage so the QA manager uses an area filter to only deploy the items that are required for testing at QA (the executables). **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Select the Deployment view and check that only the current stream is displayed. **4** Select the Pending tab. **5** In the navigation pane expand the QA stage node and select the LCL_QA_JBRNCHA_AREA03 deployment area. **6** In the content pane, from the Show list select Requests. **7** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **8** Check that the option Deploy child requests is selected. **9** Check that the Deploy Stage is set to QA. **10** Click Next. **11** To deploy the request and child task immediately, check that the option Perform deployments is set to 'as soon as possible'. **12** In the Areas(s) for deployment field check that the LCL_QA_JBRNCHA_AREA03 deployment area is selected. **13** Click Next. A summary of the deployment activity and command that will be performed is displayed. **14** Click Finish and click Close. **15** Select the History tab. **16** In the navigation pane expand the QA stage node and select the LCL_QA_JBRNCHA_AREA03 deployment area. **17** In the content pane verify that the request and child task were successfully deployed to the QA area.
The QA team tests the application.	

(Sheet 10 of 14)

Action	Procedure
The QA manager promotes the request and task to the PRE-PROD stage	QA testing has been completed successfully so Tao promotes the request and task to the PRE-PROD stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the PRE-PROD deployment area. **1** On the History tab select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that PRE-PROD is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. Dimensions CM sends an email to the release manager, Rita, notifying her that a promotion has been performed. **8** In the navigation pane select the PRE-PROD stage node. **9** In the content pane verify that the request was promoted successfully from QA to PRE-PROD.
The QA manager actions the request to its final lifecycle state	Tao closes the request. **1** On History tab select the request. **2** On the toolbar click Action. The Action wizard appears. **3** Check that the To next state field is set to CLOSED. **4** Click Finish and click OK. **5** Log out of the web client.

(Sheet 11 of 14)

Action	Procedure
The release manager deploys the request and task to the PRE-PROD deployment area	Rita reads the email, checks the Pending tab for the PRE-PROD stage on the Deployment view. Rita sees that the request and task are ready to be deployed to PRE-PROD. A build is not required at the PRE-PROD stage so the release manager uses an area filter to only deploy the items that are required at PRE-PROD (the executables). **1** Log into the web client as Rita. **2** Select the Deployment view and check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane expand the PRE-PROD stage node and select the LCL_PP_JBRNCHA_AREA03 deployment area. **5** In the content pane, from the Show list select Requests. **6** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to PRE-PROD. **9** Click Next. **10** To deploy the request and child task immediately, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the LCL_PP_JBRNCHA_AREA03 deployment area is selected. **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish and click Close. **14** Select the History tab. **15** Verify that the request and child task were successfully deployed to the PRE-PROD area.
The release team tests the application.	

(Sheet 12 of 14)

Action	Procedure
The release manager promotes the request and task to the LIVE stage	Testing has been completed successfully so Rita promotes the request and task to the LIVE stage. Deploy by Default is not enabled for the LIVE deployment area. **1** On the History tab select the request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that LIVE is selected. **4** Click Next. Rita has the privilege to deploy at the same time as the promotion but chooses not to. Do not select any deployment areas. **5** Click Next. A summary of the promotion activity and command that will be performed is displayed. **6** Click Finish. **7** In the navigation pane select the LIVE stage node. **8** In the content pane verify that the request was promoted successfully from PRE-PROD to LIVE.
The release manager deploys the request and task to the LIVE deployment area	Let's assume that it is now the regular maintenance period when the LIVE deployment area is offline. Rita checks to see what requests are ready to be deployed to the LIVE stage and uses an area filter to only deploy the executables. **1** On the History tab select the request and on the toolbar click Deploy. The Deploy wizard appears. **2** Check that the option Deploy child requests is selected. **3** Check that the Deploy Stage is set to LIVE. **4** Click Next. **5** To deploy the request and child task immediately, check that the option Perform deployments is set to 'as soon as possible'. **6** In the Areas(s) for deployment field select the LCL_LIVE_JBRNCHA_AREA03 deployment area. **7** Click Next. **8** A summary of the deployment activity and command that will be performed is displayed. **9** Click Finish and click Close.

Action	Procedure
The release manager verifies that the deployment operation was successful	Rita verifies that the deployment operation was successful. **1** In the navigation pane expand the LIVE stage node and select the LCL_LIVE_JBRNCHA_AREA03 area. **2** In the content pane verify that the request and task were successfully deployed to the LIVE area.
The release manager verifies that the executable was deployed	Rita verifies that the executable was deployed to the LIVE area. **1** In the My Current Project view expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JBRNCHA_AREA03* **2** Select the folder *Qlarius Underwriter*. **3** In the content pane verify that *LifeQuote.jar* was deployed.
End of scenario	

(Sheet 14 of 14)

Scenario Privileges

The tables below list the promotion and deployment privileges required by the users in the above scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT
	QA Manager	QA
	Release Manager	PRE-PROD

Deployment privilege	Privilege owner	Required for these areas
The DEV and SIT areas are deploy by default areas and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	QA Manager	LCL_QA_JBRNCHA_AREA03
	Release Manager	LCL_PP_JBRNCHA_AREA03 LCL_LIVE_JBRNCHA_AREA03

Scenario 3: Deploying Requests to Multiple Deployment Areas

Scenario Objective

The objective of this scenario is to deploy requests to modify the Qlarius web application. This scenario is similar to "Scenario 2: Deploying Requests to a Single Deployment Area" on page 107. The main difference is that multiple requests are deployed in a specific sequence to multiple deployment areas.

Scenario Overview

- The release manager raises an enhancement request.
- The development team lead primes two child tasks from the request and delegates them to separate developers.
- The developers modify items, deliver the modifications and relate them to the tasks, build the tasks, and capture the build outputs in Dimensions CM.
- The team lead promotes and deploys the request and tasks in a specific sequence to the SIT stage and deployment areas, and then promotes them to the QA stage.
- The QA manager deploys the request and tasks to the QA deployment areas in the same sequence and then promotes them to the PRE-PROD stage.
- The release manager deploys the request and tasks to the PRE-PROD deployment areas in the same sequence, and then promotes and deploys them to the LIVE stage and production environments.

Scenario Information

- The following stream is used: QLARIUS:JAVA_BRANCHA_STR
- There is a separation of duties between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- A build is only required at the DEV stage.
- Deployment is to multiple areas at each stage. The following deployment areas are used:

Stage	Deployment area	Deploy by Default enabled on area?	Server type	Location
DEV	LCL_DEV_JBRNCHA_AREA01	Yes	RDBMS	London
	LCL_DEV_JBRNCHA_AREA03	Yes	Web application	London
SIT	LCL_SIT_JBRNCHA_AREA01	Yes	RDBMS	London
	LCL_SIT_JBRNCHA_AREA03	Yes	Web application	London
QA	LCL_QA_JBRNCHA_AREA01	No	RDBMS	London
	LCL_QA_JBRNCHA_AREA03	No	Web application	London
PRE-PROD	LCL_PP_JBRNCHA_AREA01	No	RDBMS	London
	LCL_PP_JBRNCHA_AREA03	No	Web application	London
LIVE	LCL_LIVE_JBRNCHA_AREA01	No	RDBMS	London
	LCL_LIVE_JBRNCHA_AREA03	No	Web application	London
	LCL_LIVE_JBRNCHA_AREA04	No	RDBMS	New York
	LCL_LIVE_JBRNCHA_AREA05	No	Web application	New York

- The deployment sequence to these areas is important as the database schema must be updated before the web application code that uses it:
  - During deployment to the RDBMS areas post deployment scripts run to remotely shutdown the application servers and execute the SQL script that has been deployed to update the database schema. The update to the database is more likely to go wrong and is harder to fix than the file movement. Therefore the database is updated first so that if anything goes wrong there is less work to restore.
  - After deployment to the RDBMS server has been completed successfully, deployment continues to the web application server area. The scripts in the web application server area then re-start the servers.
- Only content relevant to the specific areas is deployed, for example, the Linux RDBMS server area receives the SQL script files and the Linux web application server area receives the Jar files.
- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 143.

Pre-Requisite

Create work areas on your local machine for the users Dinesh and Dawn, for example:

- `C:\streams\JAVA_BRANCHA_STR\dinesh`
- `C:\streams\JAVA_BRANCHA_STR\dawn`

Running this Scenario

Action	Procedure
The release manager raises an enhancement request	Changes are required to the Qlarius web application. Rita, the release manager, raises an enhancement request to manage the change. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter: *Implement support for Qlarius iPhone app* **5** In the Detailed description field enter a description. **6** On the Attributes tab, from the Severity/Priority list select a value. **7** Click Submit and then Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.
The release manager delegates the request to the team lead	Rita delegates the request to the development team lead, Ted, whose team is responsible for maintaining Qlarius. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.

(Sheet 1 of 18)

Action	Procedure
The release manager actions the request to its next state	Rita actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and then OK. The request is removed from Rita's request inbox. **4** Log out of the web client.
The development team lead primes two child tasks from the request	Ted reads the email, views the request in his Request inbox, and does some design work to see what parts of the application are affected. He primes two child tasks from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view select the Request inbox. **4** Select the request that was raised by Rita, on the toolbar click Prime, and select Task. The Prime Request dialog box appears. **5** Change the default title to *Implement support for Qlarius iPhone app (Java).* **6** Click Submit and then Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised. **7** To create a second child task repeat steps 4-6 and change the title to *Implement support for Qlarius iPhone app (sql)*
The development team lead delegates the tasks to separate developers	Ted delegates the first child task that he primed, *Implement support for Qlarius iPhone app (Java)*, to Dinesh and the second child task, *Implement support for Qlarius iPhone app (sql)*, to Dawn. Both are developers in his team. **1** Select the first child task that Ted primed and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Dinesh and click Add. The wizard closes automatically. Dimensions CM sends an email to Dinesh notifying him that a task has been added to his Request inbox. **5** Repeat steps 1-4 for the second child task and select Dawn as the candidate user.

Action	Procedure
The development team lead actions the tasks to their next state	Ted actions the tasks to their next state, UNDER WORK. **1** Select both child tasks and on the toolbar select Action. The Action multiple requests dialog box appears. **2** Click Finish. The tasks are removed from Ted's request inbox. **3** Log out of the web client.
The first developer updates their work area from the stream	Dinesh reads the email and checks his Request inbox. He does some research, identifies the item that needs to be modified, *Autoquote.java*, and updates his work area. **1** Log into the web client as Dinesh. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Change Dinesh's work area to the one that you created earlier (see the pre-requisites at the start of this scenario). **4** In the Items view, on the Dirs tab of the navigation pane, expand Qlarius Underwriter > qlarius and select *interfaces*. **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next, **7** Click Finish and then Close. Dinesh's work area is updated.
The developer modifies the item	In Dinesh's local work area edit *Autoquote.java*, which is located in this folder: ■ Qlarius Underwriter\qlarius\interfaces For the purpose of this scenario make a minor edit, for example, add a comment to the top of the item.

Action	Procedure
The developer delivers the item and relates it to the task	Dinesh delivers the modified item to Dimensions CM and relates it to the task that is delegated to him. **1** In the Items view, on the toolbar click Deliver. The Deliver to Stream wizard appears. **2** Check that the Modifications check box is selected. **3** Click Next. **4** Verify that *AutoQuote.java* is selected and click Next. **5** In the Relate to request(s) field click Select. The Select Request wizard appears. **6** From the Product name list select QLARIUS. **7** From the Type name list select TASK. **8** Click Next. **9** Select the child task that is delegated to Dinesh and click Finish. **10** In the Deliver to Stream wizard click Finish and then Close.
The developer verifies that the item was automatically deployed to the DEV areas	Deploy by default is enabled for the DEV area so when Dinesh delivered the item it was automatically deployed. He checks that the item was successfully deployed. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Select the History tab and in the navigation pane select the DEV stage node. **4** In the content pane verify that the *AutoQuote.java* was successfully deployed to both DEV deployment areas (there is a separate entry for each area): • LCL_DEV_JBRNCHA_AREA01 • LCL_DEV_JBRNCHA_AREA03 The Event Result column should display *Succeeded*.

(Sheet 4 of 18)

Action	Procedure
The developer builds the task and captures the outputs	Dinesh builds the child task in the DEV web application deployment area and captures the build outputs in Dimensions CM against the task that is delegated to him. **1** On the Requests view select the Request inbox. **2** Select the child task, on the toolbar click Build, and select Build. The Run Build wizard appears. **3** In the Build Configuration field accept the default configuration. **4** From the Build Stage list select DEV. **5** From the Build Area list select LCL_DEV_JBRNCHA_AREA03. **6** Click Next. **7** Select the option Check in build outputs automatically. This will check the build outputs into Dimensions CM. **8** To specify the request that the build outputs will be related to when they are checked into Dimensions CM, in the Specify the request(s) field click Select. The Select Request wizard appears. **9** Do the following: **a** From the Product name list select QLARIUS. **b** From the Type name list select TASK. **c** Click Next. **d** Select the child task that is delegated to Dinesh. **e** Click Finish. **10** In the Run Build wizard click Next. **11** Accept the default build option selections (none) and click Next. **12** Accept the default target selection options. **13** In the target list select *Jar Files*. **14** Click Next. A summary of the build command that will be executed is displayed. **15** Click Finish. **16** (Optional) To monitor the progress of the build click the Job <ID number> link. **17** Click Close.

(Sheet 5 of 18)

Action	Procedure
The developer verifies that the build was successful and that the outputs were captured in Dimensions CM and deployed	Dinesh verifies that the build was successful and that the outputs were captured in Dimensions CM against the task and deployed to the DEV web application deployment area. **1** To verify that the build was successful, in the Deployment view select the History tab. **2** In the navigation pane expand the DEV stage node and select LCL_DEV_JBRNCHA_AREA03. **3** In the content pane verify that the Event Result column displays *Succeeded* for the following objects: • Autoquote.jar (the event type is Collect) • QLARIUS:JAVA_BRANCHA_STR (the event type is Build) **4** Open *Autoquote.jar*, make a note of the item revision, and click Cancel. **5** To verify that the build outputs were captured in Dimensions CM against the task, in the Requests view select the Request inbox and open the task. The Open Request dialog box appears. **6** Select the Relationships tab and from the Related object class list select Items. A list of all the build outputs that were captured in Dimensions CM against the task is displayed. **7** Click Cancel. **8** To browse the DEV web application deployment area, in the My Current Project view expand *Deployment Areas > DEV Stage > LCL_DEV_JBRNCHA_AREA03* **9** Select the folder *Qlarius Underwriter*. **10** In the content pane verify that latest revision of *AutoQuote.jar* was deployed.
The developer delegates the task to the team lead for peer review	Dinesh delegates the task to Ted for peer review. **1** In the Requests view select the Request inbox. **2** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.

Action	Procedure
The developer actions the task to its next state	Dinesh actions the task that is delegated to him to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Dinesh's request inbox. **7** Log out of the web client.
Log into the web client as Dawn and perform the following, similar, actions: **1** Switch to the stream QLARIUS:JAVA_BRANCHA_STR. **2** Set the work area for Dawn. **3** Update Dawn's local work area from the stream. **4** Edit the item *qlarius.sql* located in *Qlarius Underwriter\qlarius\sql*. Note: Dawn builds and tests the sql file locally. **5** Deliver the modified item back to the stream and relate it to the task that was delegated to Dawn. Make a note of the latest item revision. **6** In the Deployment view verify that *qlarius.sql* was deployed successfully to the LCL_DEV_JBRNCHA_AREA01 deployment area. **7** In the My Current Project view browse the DEV RDBMS deployment area and verify that the latest revision of *qlarius.sql* was deployed to LCL_DEV_JBRNCHA_AREA01 > *Qlarius Underwriter* > *qlarius* > *sql*. **8** Delegate the task to Ted for peer review. **9** Action the task to PEER REVIEW. **10** Log out of the web client.	

(Sheet 7 of 18)

Action	Procedure
The team lead does a peer review and actions the tasks to their final state	Ted has read his emails, seen the tasks in his Request inbox, done a peer review of the items that Dinesh and Dawn modified, and is satisfied with the changes that they made. He actions the tasks to their final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox, select both child tasks, and on the toolbar click Action. The Action wizard appears. **3** Click Finish. The child tasks are removed from Ted's Request inbox.
The team lead promotes and deploys the request and tasks to the SIT stage and RDBMS server area	To perform system integration testing, Ted promotes and deploys the request and the child tasks to the SIT stage and deployment areas in the following order: ■ RDBMS server area ■ Web application server area Deploy by default is enabled for the SIT areas. A build is not required at the SIT stage so the team lead uses an area filter to only deploy the items that are required for testing at SIT (the executables). **1** Select the parent request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. This promotes the child task with its parent request. **3** In the Next stage field check that SIT is selected. **4** Click Next. **5** To deploy immediately, check that the option Perform deployments is set to 'as soon as possible'. **6** In the Areas(s) for deployment field select the SIT LCL_SIT_JBRNCHA_AREA01 deployment area (the RDBMS server). Note: If both areas are selected, deselect SIT LCL_SIT_JBRNCHA_AREA03. **7** Click Next. A summary of the promotion and deployment activities and command that will be performed is displayed. **8** Click Finish.

(Sheet 8 of 18)

Action	Procedure
The team lead verifies that promotion and deployment operations were successful	Ted verifies that promotion and deployment operations were successful and that the sqlemail file was deployed to the SIT RDBMS deployment area. **1** Select the Deployment view and check that only the current stream is displayed. **2** Select the History tab and in the navigation pane select the SIT stage node. **3** In the content pane verify that the request was promoted successfully from DEV to SIT. **4** In the navigation pane expand the SIT stage node and select the LCL_SIT_JBRNCHA_AREA01 deployment area. **5** In the content pane verify that the request and child tasks were deployed successfully to the SIT RDBMS deployment area. **6** To browse the SIT RDBMS deployment area, in the My Current Project view expand *Deployment Areas > SIT Stage > LCL_SIT_JBRNCHA_AREA01.* **7** Expand *Qlarius Underwriter > qlarius > sql.* **8** In the content pane verify that the latest revision of *qlarius.sql* was deployed.
The team lead deploys the request and tasks to the SIT web application server area	Deployment to the RDBMS server area has been completed successfully so Ted now deploys the request and tasks to the SIT web application server area. **1** In the Deployment view select the Pending tab and in the navigation pane select the SIT stage node. **2** In the content pane, from the Show list select Requests. **3** In the content pane select the request and on the toolbar click Deploy. The Deploy wizard appears. **4** Check that the option Deploy child requests is selected. **5** In the Next stage field check that SIT is selected. **6** Click Next. **7** To deploy immediately, check that the option Perform deployments is set to 'as soon as possible'. **8** In the Areas(s) for deployment field check that the LCL_SIT_JBRNCHA_AREA03 deployment area (the SIT web application server) is selected. **9** Click Next. A summary of the deployment activity and command that will be performed is displayed. **10** Click Finish.

(Sheet 9 of 18)

Action	Procedure
The team lead verifies that promotion and deployment operations were successful	Ted verifies that promotion and deployment operations were successful and that the .jar file was deployed to the SIT web application deployment area. **1** Select the History tab. **2** In the navigation pane select the LCL_SIT_JBRNCHA_AREA03 deployment area. **3** In the content pane verify that the request and child tasks were deployed successfully to the SIT web application deployment area. **4** To browse the SIT web application deployment area, in the My Current Project view expand *Deployment Areas > SIT Stage > LCL_SIT_JBRNCHA_AREA03.* **5** Select the folder *Qlarius Underwri*ter. **6** In the content pane verify that the latest revision of *AutoQuote.jar* was deployed.
Ted performs system integration testing.	
The team lead promotes the request and tasks to the QA stage	System integration testing has been completed successfully so Ted promotes the request and tasks to the QA stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the QA deployment area. **1** On the Requests view, in the Request inbox, select the request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that QA is selected. **4** Click Next. Deploy by Default is not enabled so no deploy options are available. **5** Click Next. A summary of the promotion activity and command that will be performed is displayed. **6** Click Finish. Dimensions CM sends an email to the QA manager, Tao, notifying her that a promotion has been performed. **7** In the Deployment view select the History tab and in the navigation pane select the QA stage node. **8** In the content pane verify that the request was promoted successfully from SIT to QA.

Action	Procedure
The team lead actions the request to its next state	Ted actions the request to its next lifecycle state, IN TEST, so that QA can test the application. **1** On the History tab select the request and on the toolbar click Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to IN TEST. **3** Click Next. **4** In the Details of solution given field enter a solution. **5** Click Finish. Note: The task is removed from Ted's request inbox. **6** Log out of the web client.

(Sheet 11 of 18)

Action	Procedure
The QA manager deploys the request and tasks to the QA deployment areas	Tao reads the email, checks her Request inbox, and deploys the request and tasks to the QA deployment areas in the same sequence: ■ RDBMS server area ■ Web application server area A build is not required at the QA stage so the QA manager uses an area filter to only deploy the items that are required for testing at QA (the executables). **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Select the Deployment view and check that only the current stream is displayed. **4** Select the Pending tab. **5** In the navigation pane expand the QA stage node and select the LCL_QA_JBRNCHA_AREA01 deployment area (the RDBMS server). **6** In the content pane, from the Show list select Requests. **7** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **8** Check that the option Deploy child requests is selected. **9** Check that the Deploy Stage is set to QA. **10** Click Next. **11** To deploy the request and child tasks immediately, check that the option Perform deployments is set to 'as soon as possible'. **12** In the Areas(s) for deployment field check that the LCL_QA_JBRNCHA_AREA01 deployment area (the RDBMS server) is selected. **13** Click Next. A summary of the deployment activity and command that will be performed is displayed. **14** Click Finish. **15** Select the History tab. **16** In the content pane verify that the request and child tasks were successfully deployed to the QA area LCL_QA_JBRNCHA_AREA01. **17** Repeat steps 4-16 for the QA web application server area: LCL_QA_JBRNCHA_AREA03.
The QA team performs their tests.	

(Sheet 12 of 18)

Action	Procedure
The QA manager actions the request to its final lifecycle state.	QA testing has been completed successfully so Tao closes the enhancement request. **1** On the History tab select the request and on the toolbar click Action. The Action wizard appears. **2** Check that the To next state field is set to CLOSED. **3** Click Finish and click OK. Note: The request is removed from Tao's request inbox.
The QA manager promotes the request and tasks to the PRE-PROD stage	Tao promotes the request and tasks to the PRE-PROD stage. Deploy by Default is not enabled so the request and tasks cannot be automatically deployed to the PRE-PROD deployment area. **1** On the History tab select the request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that PRE-PROD is selected. **4** Click Next. Deploy by Default is not enabled so no deploy options are available. **5** Click Next. A summary of the promotion activity and command that will be performed is displayed. **6** Click Finish. Dimensions CM sends an email to the release manager, Rita, notifying her that a promotion has been performed. **7** In the navigation pane select the PRE-PROD stage node. **8** In the content pane verify that the request was promoted successfully from QA to PRE-PROD. **9** Log out of the web client.

(Sheet 13 of 18)

Action	Procedure
The release manager deploys the request and tasks to the PRE-PROD deployment areas	Rita, the release manager, reads the email and checks the Pending tab for the PRE-PROD stage on the Deployment view. Rita sees that the request and tasks are ready to be deployed to PRE-PROD. The deployment sequence is the same as in the previous stages. A build is not required at the PRE-PROD stage so the release manager uses an area filter to only deploy the items that are required at PRE-PROD (the executables). **1** Log into the web client as Rita. **2** Select the Deployment view and check that only the current stream is displayed. **3** Select the Pending tab, in the navigation pane expand the PRE-PROD stage node, and select the LCL_PP_JBRNCHA_AREA01 deployment area (the RDBMS server). **4** In the content pane, from the Show list select Requests. **5** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **6** Check that the option Deploy child requests is selected. **7** Check that the Deploy Stage is set to PRE-PROD. **8** Click Next. **9** To deploy the request and child tasks immediately, check that the option Perform deployments is set to 'as soon as possible'. **10** In the Areas(s) for deployment field check that the LCL_PP_JBRNCHA_AREA01 deployment area (the RDBMS server) is selected. **11** Click Next. A summary of the deployment activity and command that will be performed is displayed. **12** Click Finish. **13** Select the History tab. **14** In the content pane verify that the request and child tasks were successfully deployed to the PRE-PROD LCL_PP_JBRNCHA_AREA01 deployment area. **15** Repeat steps 3-14 for the PRE-PROD web application server area: LCL_PP_JBRNCHA_AREA03.
The release team performs their tests.	

(Sheet 14 of 18)

Action	Procedure
The release manager promotes the request and tasks to the LIVE stage	Testing has been completed so Rita promotes the request and tasks to the LIVE stage. Deploy by Default is not enabled for the LIVE deployment area. **1** On the History tab select the request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that LIVE is selected. **4** Click Next. Rita has the privilege to deploy at the same time as the promotion but chooses not. Do not select any deployment areas. **5** Click Next. A summary of the promotion activity and command that will be performed is displayed. **6** Click Finish. **7** In the navigation pane select the LIVE stage node. **8** In the content pane verify that the request was promoted successfully from PRE-PROD to LIVE.
The request and tasks are now ready to be deployed to the company's live production environments in London and New York. Deployment normally takes place during the regular maintenance period when the areas are offline. Rita does the following: ■ Schedules the deployment to the New York production environment at midnight local time. ■ Manually deploys to the London production environment at midnight local time. London is ahead of New York so if there is a problem with the deployment she has time to fix it. Note: Rita deploys to the LIVE deployment areas in the same sequence as in the previous stages.	

(Sheet 15 of 18)

Action	Procedure
The release manager schedules a deployment to the live production environment in New York	Rita schedules the deployment of the request and tasks to the live production environment in New York at midnight local time. **1** In the navigation pane expand the LIVE stage node and select the RDBMS deployment area in New York: LCL_LIVE_JBRNCHA_AREA04 **2** On the Pending tab select the request and on the toolbar click Deploy. The Deploy wizard appears. Note: You can also deploy the request from the History tab. **3** Check that the option Deploy child requests is selected. **4** Check that the Deploy Stage is set to LIVE. **5** Click Next. **6** To schedule the deployment select the option 'at specified time'. **7** Click the Calendar button. **8** Select today's date at midnight. Tip: To test this scenario, schedule the deployment to execute soon, for example, in 15 minutes. **9** In the Areas(s) for deployment field check that the LCL_LIVE_JBRNCHA_AREA04 deployment area is selected. **10** Click Next. **11** A summary of the deployment activity and command that will be performed, including the date and time, is displayed. **12** Click Finish. **13** On the Queue tab verify that the request and tasks are queued and waiting to be deployed. **14** Repeat steps 1-13 for the web application deployment area in New York, LCL_LIVE_JBRNCHA_AREA05. You must schedule the deployment a few minutes later than the deployment to the RDBMS area.

(Sheet 16 of 18)

Action	Procedure
The release manager deploys the request and the tasks to the LIVE RDBMS deployment area in London	Let's assume that it is now midnight in London and the local live production environments are offline. Rita checks to see what requests are ready to be deployed to the LIVE stage. She uses an area filter to only deploy the executables. **1** In the navigation pane, in the LIVE stage node, select the RDBMS deployment area in London: LCL_LIVE_JBRNCHA_AREA01 **2** On the Pending tab select the request and on the toolbar click Deploy. The Deploy wizard appears. **3** Check that the option Deploy child requests is selected. **4** Check that the Deploy Stage is set to LIVE. **5** Click Next. **6** To deploy the request and child task immediately, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field check that the LCL_LIVE_JBRNCHA_AREA01 deployment area is selected. **8** Click Next. **9** A summary of the deployment activity and command that will be performed, including the date and time, is displayed. **10** Click Finish.
The release manager verifies that deployment to the RDBMS area was successful	Rita verifies that deployment operation was successful and that the sql file was deployed to the LIVE RDBMS deployment area in London. **1** Select the History tab. **2** In the content pane verify that the request and tasks were successfully deployed to the LIVE area. **3** To browse the LIVE deployment area, in the My Current Project view expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JBRNCHA_AREA01* **4** Expand *Qlarius Underwriter* > *qlarius* > *sql*. **5** In the content pane verify that the latest revision of *qlarius.sq* was deployed.

(Sheet 17 of 18)

Action	Procedure
The release manager deploys the request and tasks to the LIVE web application deployment area in London	The deployment to the RDBMS area was successful so Rita can now deploy the request and tasks to the LIVE web application deployment area in London. **1** In the Deployment view, in the LIVE stage node in the navigation pane select the web application deployment area in London: LCL_LIVE_JBRNCHA_AREA03 **2** On the Pending tab select the request and on the toolbar click Deploy. The Deploy wizard appears. **3** Check that the option Deploy child requests is selected. **4** Check that the Deploy Stage is set to LIVE. **5** Click Next. **6** To deploy the request and child task immediately, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field check that the LCL_LIVE_JBRNCHA_AREA03 deployment area is selected. **8** Click Next. **9** A summary of the deployment activity and command that will be performed, including the date and time, is displayed. **10** Click Finish.
The release manager verifies that deployment to the web application area was successful	Rita verifies that deployment operation was successful and that the jar file was deployed to the LIVE RDBMS deployment area in London. **1** Select the History tab. **2** In the content pane verify that the request and tasks were successfully deployed to the LIVE area. **3** To browse the LIVE deployment area, in the My Current Project view expand *Deployment Areas > LIVE Stage > LCL_LIVE_JBRNCHA_AREA03* **4** Select the folder *Qlarius Underwriter*. **5** In the content pane verify that the latest revision of *AutoQuote.jar* was deployed.
Rita emails the New York office to advise them that a deployment has been scheduled for midnight local time and that the deployment in London was successful. Rita goes to sleep and in the morning verifies that the deployment to the production environment in New York was successful. The enhancement to the Qlarius web application is now live across all production environments.	
End of scenario	

(Sheet 18 of 18)

Scenario Privileges

The tables below list the promotion and deployment privileges required by each user in the scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT
	QA Manager	QA
	Release Manager	PRE-PROD

Deployment privilege	Privilege owner	Required for these areas
The DEV area is a deploy by default area and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	Team Lead	LCL_SIT_JBRNCHA_AREA01 LCL_SIT_JBRNCHA_AREA03
	QA Manager	LCL_QA_JBRNCHA_AREA01 LCL_QA_JBRNCHA_AREA03
	Release Manager	LCL_PP_JBRNCHA_AREA01 LCL_PP_JBRNCHA_AREA03 LCL_LIVE_JBRNCHA_AREA01 LCL_LIVE_JBRNCHA_AREA03 LCL_LIVE_JBRNCHA_AREA04 LCL_LIVE_JBRNCHA_AREA05

Scenario 4: Deploying Refactoring Changes

Scenario Objectives

The objective of this scenario is to deploy refactoring changes to the corporate web site of Qlarius Health Insurance.

Scenario Overview

- The release manager raises enhancement requests.
- The development team lead primes a child task from each request.
- A web developer makes the refactoring changes, delivers the modifications, and relates them to the tasks.
- The team lead promotes and deploys the requests and tasks to the SIT stage and deployment area, and then promotes them to the QA stage.
- The QA manager deploys the requests and tasks to the QA deployment area and then promotes them to the PRE-PROD stage.
- The release manager deploys the requests and tasks to the PRE-PROD deployment area and then promotes and deploys them to the LIVE stage and production environments.

Scenario Information

- The following scenario is used: QLARIUS:JAVA_BRANCHA_STR
- No builds are required as only a text file is modified.
- There is a division of responsibilities between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- This scenario uses the following web application deployment areas:

Stage	Deployment area	Deploy by Default enabled for area?
DEV	LCL_DEV_JBRNCHA_AREA03	Yes
SIT	LCL_SIT_JBRNCHA_AREA03	Yes
QA	LCL_QA_JBRNCHA_AREA03	No
PRE-PROD	LCL_PP_JBRNCHA_AREA03	No
LIVE	LCL_LIVE_JBRNCHA_AREA03	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 161.

Pre-Requisites

1 Create a work area on your local machine for the user Wendy, for example:

```
C:\streams\JAVA_BRANCHA_STR\wendy
```

2 Log into the web client as a user that has the privileges to promote and deploy baselines to any stage and area, for example, the tool administrator, typically *dmsys*.

3 Take a tip baseline of the stream JAVA_BRANCHA_STR.

4 To deploy the files in the stream to all deployment areas, promote and deploy the baseline as follows:

a Select the baseline and on the toolbar click Promote.

b In the Next stage field check that SIT is selected.

c Click Next.

d To deploy now, check that the option Perform deployments is set to 'as soon as possible'.

e In the Areas(s) for deployment field check that the LCL_SIT_JBRNCHA_AREA03 deployment area is selected.

f Click Next.

g A summary of the promotion and deployment activities and command that will be performed is displayed.

h Click Finish.

i Repeat steps 'a' to 'h' for the other stages and deployment areas:

- QA: LCL_QA_JBRNCHA_AREA03
- PRE-PROD: LCL_PRE-PROD_JBRNCHA_AREA03
- LIVE: LCL_LIVE_JBRNCHA_AREA03

5 Log out of the web client.

Running this Scenario

Action	Procedure
The release manager raises enhancement requests	Refactoring changes are required to the corporate web site of Qlarius Health Insurance. Rita, the release manage, raises two enhancement requests to manage the change: ■ The first request is a refactoring change to create a *resources* directory and move the file *main.css* from the *website* directory to the new directory. ■ The second request is an enhancement to *main.css*. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter: *Create resources directory and move CSS file* **5** In the Detailed description field enter a description. **6** On the Attributes tab, from the Severity/Priority list select a value. **7** Click Submit and click Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_n By default the request is at the DEV stage when it is raised. **8** Repeat steps 3-7 to create the second request and use the title *Modify CSS file.*
The release manager delegates the request to the team lead	Rita delegates the requests to the development team lead, Ted, whose team is responsible for maintaining the web site. **1** Select both requests and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that requests have been added to his Request inbox.

(Sheet 1 of 15)

Action	Procedure
The release manager actions the requests to their next state	Rita actions the requests to their next state, UNDER WORK. **1** Select the requests and on the toolbar select Action. The Action multiple requests dialog box appears. **2** Click Finish. The requests are removed from Rita's request inbox. **3** Log out of the web client.
The development team lead primes a child task from each request	Ted reads the email, views the request in his Request inbox, and primes a separate child task from each request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view select the Request inbox. **4** Select the first request that was raised by Rita, *Create resources directory and move CSS file*. **5** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **6** Click Submit and then Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised. **7** To prime a child task from the second request, *Modify CSS file,* repeat steps 4-6.
The development team lead delegates the tasks to a web developer	Ted delegates both tasks to a web developer, Wendy. **1** Select both tasks and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Wendy and click Add. The wizard closes automatically. Dimensions CM sends an email to Wendy notifying her that requests have been added to her Request inbox.

(Sheet 2 of 15)

Action	Procedure
The development team lead actions the tasks to their next state	Ted actions the tasks to their next state, UNDER WORK. **1** Select both child tasks and on the toolbar select Action. The Action multiple requests dialog box appears. **2** Click Finish. The tasks are removed from Ted's request inbox. **3** Log out of the web client.
The web developer updates her local work area	Wendy, the web developer, reads the emails and checks her Request inbox. She updates her local work area from the *website* folder. **1** Log into the web client as Wendy. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Change Wendy's work area to the one that you created earlier (see the pre-requisites at the start of this scenario). **4** In the Items view, in the navigation pane select the Dirs tab, expand *Qlarius Underwriter* and select *website*. **5** On the toolbar click Update. The Update from Stream dialog box appears. **6** Click Next and then Finish. **7** Click Close.
The web developer makes some refactoring changes	In Wendy's local work area do the following: **1** In the *website* folder create a new subfolder called *resources*. **2** Move *main.css* from the *website* folder to the *resources* folder.

(Sheet 3 of 15)

Action	Procedure
The web developer delivers the refactoring changes	Wendy delivers the refactoring changes to Dimensions CM and relates them to the first task. **1** In the web client, on the navigation pane, select the *website* folder (on the Dirs tab of the Items view). **2** On the toolbar click Deliver. The Deliver to Stream dialog box appears. **3** Check that the option Moves/Renames is selected. **4** Click Next. The following changes types should be selected: • Qlarius Underwriter\website\main.css (Deletion) • Qlarius Underwriter\website\resources (Addition) • Qlarius Underwriter\website\resources\main.css (Addition) **5** Click Next. **6** In the Relate to Request(s) field click Select. The Select Request wizard appears. **7** From the Product name list select QLARIUS. **8** From the Type name list select TASK. **9** Click Next. **10** Select the task *Create resources directory and move CSS file* and click Finish. **11** In the Deliver to Stream dialog box click Finish. **12** Click Close. The refactoring changes are delivered to Dimensions CM.

(Sheet 4 of 15)

Action	Procedure
The web developer verifies that the refactoring changes were deployed	Deploy by default is enabled for the DEV web application deployment area so when Wendy delivered the refactoring changes they were automatically deployed. She checks that deployment was successful. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Select the History tab. **4** In the navigation pane expand the DEV stage node and select the LCL_DEV_JBRNCHA_AREA03 web application deployment area. **5** In the content pane verify that the *main.css* was successfully deployed. The Event Result column should display *Succeeded*. **6** To browse the DEV web application deployment area, in the My Current Project view expand *Deployment Areas* > *DEV Stage* > *LCL_DEV_JBRNCHA_AREA03* > *Qlarius Underwriter* > *website*. In the navigation pane verify that there is a folder called *resources*. **7** Select the *resources* directory and in the content pane verify that *main.css* was deployed.
The web developer modifies the file	In Wendy's local work area modify *main.css* in the *resources* directory. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the file.

(Sheet 5 of 15)

Action	Procedure
The web developer delivers the modification to Dimensions CM	Wendy delivers the modified file to Dimensions CM relates it to the second task. **1** In the web client, in the Items view, on the Dirs tab expand *Qlarius Underwriter* and select the *resources* folder. **2** On the toolbar click Deliver. The Deliver to Stream dialog box appears. **3** Click Next. **4** Check that *main.css* is selected and click Next. **5** In the Relate to Request(s) field click select. The Select Request wizard appears. **6** From the Product name list select QLARIUS. **7** From the Type name list select TASK. **8** Click Next. **9** Select the task *Modify CSS file* and click Finish. **10** In the Deliver to Stream dialog box click Finish. **11** Click Close. Make a note of the latest revision of *main.css*.
The web developer verifies that the modification was deployed	Wendy checks that deployment was successful. **1** Select the Deployment view. **2** Select the History tab. **3** In the navigation pane expand the DEV stage node and select the LCL_DEV_JBRNCHA_AREA03 web application deployment area. **4** In the content pane verify that the *main.css* was successfully deployed a second time. The Event Result column should display *Succeeded*. **5** To browse the DEV web application deployment area, in the My Current Project view expand *Deployment Areas > DEV Stage > LCL_DEV_JBRNCHA_AREA03> Qlarius Underwriter > website > resources.* **6** Select the *resources* directory and in the content pane verify that the latest revision of *main.css* is deployed.

Action	Procedure
The web developer delegates the tasks to the team lead for peer review	Wendy delegates the tasks to Ted for peer review. **1** In the Requests view select the Request inbox. **2** Select both tasks and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that tasks have been added to his Request inbox.
The web developer actions the tasks to their next state	Wendy actions the tasks to their next state, PEER REVIEW. **1** Select the first child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Wendy's Request inbox. **7** Repeat steps 1-6 for the second child task. **8** Log out of the web client.
The team lead does a peer review and actions the tasks to their final state	Ted has read his emails, seen the tasks in his Request inbox, and done a peer review of the refactoring changes. He actions both tasks to their final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox, select both tasks, and on the toolbar click Action. The Action wizard appears. **3** Click Finish. The child tasks are removed from Ted's Request inbox.

(Sheet 7 of 15)

Action	Procedure
The team lead promotes and deploys the requests and tasks to the SIT stage	To perform system integration testing, Ted promotes and deploys the requests and tasks to the SIT stage and web application deployment area. Ted promotes both requests in the same operation and lets Dimensions CM automatically deploy the refactoring changes in the correct sequence. Dimensions CM does the following: ■ Creates the *resources* directory and moves *main.css* from the *website* directory to the *resources* directory. ■ Deploys the latest revision of *main.css* to the *resources* directory. **Note**: If you select multiple requests, or a request with child requests, Dimensions CM automatically resolves the refactoring changes. This applies to both manual and automatic (deploy by default) deployments. **1** In the Request view select both parent requests. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. This promotes and deploys the child tasks with their parent requests. **4** In the Next stage field check that SIT is selected. **5** Click Next. **6** To deploy now, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field select the SIT LCL_SIT_JBRNCHA_AREA03 deployment area. Note: If both areas are selected, deselect SIT LCL_SIT_JBRNCHA_AREA01. **8** Click Next. A summary of the promotion and deployment activities and command that will be performed is displayed. **9** Click Finish.

(Sheet 8 of 15)

Action	Procedure
The team lead verifies that promotion and deployment operations were successful	Ted verifies that promotion and deployment operations were successful and that the refactoring changes were deployed to the SIT web application deployment area. **1** Select the Deployment view and check that only the current stream is displayed. **2** Select the History tab and in the navigation pane select the SIT stage node. **3** In the content pane verify that both requests were promoted successfully from DEV to SIT. **4** In the navigation pane expand the SIT stage node and select the LCL_SIT_JBRNCHA_AREA03 deployment area. **5** In the content pane verify that all the requests and child tasks were deployed successfully. **6** To browse the SIT web application deployment area, in the My Current Project view expand *Deployment Areas* > *SIT Stage* > *LCL_SIT_JBRNCHA_AREA03* > *Qlarius Underwriter* > *website*. In the navigation pane verify that there is a folder called *resources*. **7** Select the *resources* directory and in the content pane verify that the latest revision of *main.css* is deployed.
Ted performs system integration testing.	
The team lead promotes the requests and tasks to the QA stage	System integration testing has been completed successfully so Ted promotes the requests and tasks to the QA stage. Deploy by Default is not enabled so the requests and tasks cannot be automatically deployed to the QA deployment area. **1** In the Requests view select the requests. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that QA is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. Dimensions CM sends an email to the QA manager, Tao, notifying her that a promotion has been performed. **8** In the Deployment view select the History tab and in the navigation pane select the QA stage node. **9** In the content pane verify that both requests were promoted successfully from SIT to QA.

Action	Procedure
The team lead actions the requests to their next state	Ted actions the requests to their next lifecycle state, IN TEST, so that QA can test the web site. **1** On the History tab select the first request and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to IN TEST. **3** Click Next. **4** In the Details of solution given field enter a description. **5** Click Finish. Note: The request is removed from Ted's Request inbox. **6** Repeat steps 1-5 for the second request. **7** Log out of the web client.

(Sheet 10 of 15)

Action	Procedure
The QA manager deploys the requests and tasks to the QA deployment area	Tao reads the email and checks her Request inbox. She deploys both requests in the same operation and lets Dimensions CM automatically deploy the refactoring changes in the correct sequence. **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Select the Deployment view and check that the current stream is displayed. **4** Select the Pending tab. **5** In the navigation pane expand the QA stage node and select the LCL_QA_JBRNCHA_AREA03 deployment area. **6** In the content pane, from the Show list select Requests. **7** Select both requests and on the toolbar click Deploy. The Deploy wizard appears. **8** Check that the option Deploy child requests is selected. **9** Check that the Deploy Stage is set to QA. **10** Click Next. **11** To deploy the requests now, check that the option Perform deployments is set to 'as soon as possible'. **12** In the Areas(s) for deployment field check that the following QA deployment area is selected: LCL_QA_JBRNCHA_AREA03 **13** Click Next. A summary of the deployment activity and command that will be performed is displayed. **14** Click Finish. **15** Select the History tab. **16** In the content pane verify that the requests and child tasks were successfully deployed to the QA area LCL_QA_JBRNCHA_AREA03.
The QA manger verifies that the refactoring changes were deployed	Tao verifies that the refactoring changes were deployed to the QA web application deployment area. **1** In the My Current Project view expand *Deployment Areas* > *QA Stage* > *LCL_QA_JBRNCHA_AREA03* > *Qlarius Underwriter* > *website*. In the navigation pane verify that there is a folder called *resources*. **2** Select the *resources* directory and in the content pane verify that the latest revision of *main.css* is deployed.
The QA team performs their tests.	

(Sheet 11 of 15)

Action	Procedure
The QA manager actions the requests to their final lifecycle state.	QA testing has been completed successfully so Tao closes the enhancement requests. **1** On the History tab select both requests and on the toolbar click Action. The Action wizard appears. **2** Click Finish. Note: The request are removed from Tao's request inbox.
The QA manager promotes the requests and tasks to the PRE-PROD stage.	QA testing has been completed successfully so Tao promotes the requests and tasks to the PRE-PROD stage. Deploy by Default is not enabled so the requests and tasks cannot be automatically deployed to the PRE-PROD deployment area. **1** On the History tab select the requests. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that PRE-PROD is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. Dimensions CM sends an email to Rita, the release manager, notifying her that a promotion has been performed. **8** In the navigation pane select the PRE-PROD stage node. **9** In the content pane verify that both requests were promoted successfully from QA to PRE-PROD. **10** Log out of the web client.

Action	Procedure
The release manager deploys the requests and tasks to the PRE-PROD deployment area	Rita reads the email and checks her Request inbox. She deploys both requests in the same operation and lets Dimensions CM automatically deploy the refactoring changes in the correct sequence. **1** Log into the web client as Rita. **2** Select the Deployment view and check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane expand the PRE-PROD stage node and select the LCL_PP_JBRNCHA_AREA03 deployment area. **5** In the content pane, from the Show list select Requests. **6** Select both requests and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to PRE-PROD. **9** Click Next. **10** To deploy the requests now, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the following PRE-PROD deployment area is selected: LCL_PP_JBRNCHA_AREA03 **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish. **14** Select the History tab. **15** In the content pane verify that the requests and child tasks were successfully deployed to the QA area LCL_PP_JBRNCHA_AREA03.
The release manger verifies that the refactoring changes were deployed	Rita verifies that the refactoring changes were deployed to the PRE-PROD web application deployment area. **1** In the My Current Project view expand *Deployment Areas* > *PRE-PROD Stage* > *LCL_PP_JBRNCHA_AREA03* > *Qlarius Underwriter* > *website*. In the navigation pane verify that there is a folder called *resources*. **2** Select the *resources* directory and in the content pane verify that the latest revision of *main.css* is deployed.
The release team performs their tests.	

(Sheet 13 of 15)

Action	Procedure
The release manager promotes the requests and tasks to the LIVE stage	Testing has been completed successfully so Rita promotes the requests and tasks to the LIVE stage. Deploy by Default is not enabled so the requests and tasks cannot be automatically deployed to the PRE-PROD deployment area. **1** In the Deployment view, in the navigation pane, select the LCL_PP_JBRNCHA_AREA03 deployment area. **2** Select the History tab and select the requests. **3** On the toolbar click Promote. The Promote wizard appears. **4** Check that the option Promote child requests is selected. **5** In the Next stage field check that LIVE is selected. **6** Click Next. Rita has the privilege to deploy at the same time as the promotion but chooses not. Do not select any deployment areas. **7** Click Next. A summary of the promotion activity and command that will be performed is displayed. **8** Click Finish. **9** In the Deployment view select the History tab and in the navigation pane select the LIVE stage node. **10** In the content pane verify that both requests were promoted successfully from PRE-PROD to LIVE.

(Sheet 14 of 15)

Action	Procedure
The release manager deploys the requests and tasks to the LIVE deployment area	Let's assume that it is now the regular maintenance period when the LIVE web application deployment area is offline. Rita checks to see what requests are ready to be deployed to the LIVE stage. Rita deploys both requests in the same operation and lets Dimensions CM automatically deploy the refactoring changes in the correct sequence. **1** Select the Pending tab. **2** In the navigation pane select the LCL_LIVE_JBRNCHA_AREA03 deployment area. **3** In the content pane, from the Show list select Requests. **4** Select both requests and on the toolbar click Deploy. The Deploy wizard appears. **5** Check that the option Deploy child requests is selected. **6** Check that the Deploy Stage is set to LIVE. **7** Click Next. **8** To deploy the requests now, check that the option Perform deployments is set to 'as soon as possible'. **9** In the Areas(s) for deployment field check that the following LIVE deployment area is selected: LCL_LIVE_JBRNCHA_AREA03 **10** Click Next. A summary of the deployment activity and command that will be performed is displayed. **11** Click Finish. **12** Select the History tab. **13** In the content pane verify that the requests and child tasks were successfully deployed to the QA area LCL_LIVE_JBRNCHA_AREA03.
The release manger verifies that the refactoring changes were deployed	Rita verifies that the refactoring changes were deployed to the LIVE web application deployment area. **1** In the My Current Project view expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JBRNCHA_AREA03* > *Qlarius Underwriter* > *website*. In the navigation pane verify that there is a folder called *resources*. **2** Select the *resources* directory and in the content pane verify that the latest revision of *main.css* is deployed.
End of scenario	

(Sheet 15 of 15)

Scenario Privileges

The tables below list the promotion and deployment privileges required by each user in the scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT
	QA Manager	QA
	Release Manager	PRE-PROD

Deployment privilege	Privilege owner	Required for these areas
The SIT and LIVE areas are deploy by default areas and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	QA Manager	LCL_QA_JBRNCHA_AREA03
	Release Manager	LCL_PP_JBRNCHA_AREA03

Scenario 5: Rolling Back a Deployment

Scenario Objective

There is a problem with the corporate web site of Qlarius Health Insurance. The objective of the scenario is to rollback to the previous version, create a fix, and then deploy the fix to the live environment.

For example, let's assume that revision 2 of a file is currently deployed to the LIVE deployment area. The rollback operation redeploys revision 1 to LIVE. The fix creates revision 3, which is promoted and deployed up the GSL to LIVE and replaces revision 1.

Scenario Overview

- The release manager rolls back the request from the LIVE deployment area, demotes the request to the PRE-PROD stage, and raises a change request to track the defect.
- The development team lead primes a child task from the request.
- A web developer fixes and delivers an item, and relates it to the task.
- The team lead promotes and deploys the request and task to the SIT stage and deployment area, and then promotes them to the QA stage.
- The QA manager deploys the request and task to the QA deployment area and then promotes them to the PRE-PROD stage.
- The release manager deploys the request and task to the PRE-PROD deployment area, and then promotes and deploys them to the LIVE stage and production environment.

Scenario Information

- The following stream is used: QLARIUS:MAINLINE_JAVA_STR
- There is a division of responsibilities between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- No build is required at any stage as only a text file is changed.
- This scenario uses the following deployment areas:

Stage	Deployment area	Deploy by Default enabled for area?
DEV	LCL_DEV_JMAIN_AREA01	Yes
SIT	LCL_SIT_JMAIN_AREA01	Yes
QA	LCL_QA_JMAIN_AREA01	No
PRE-PROD	LCL_PP_JMAIN_AREA01	No
LIVE	LCL_LIVE_JMAIN_AREA01	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 178.

Pre-Requisites

1 To run this scenario you must have successfully completed "Scenario 1: Basic Request Deployment" on page 92.

2 Log into the web client as any user.

3 Switch to the stream QLARIUS:MAINLINE_JAVA_STR.

4 On the Request view select Catalog.

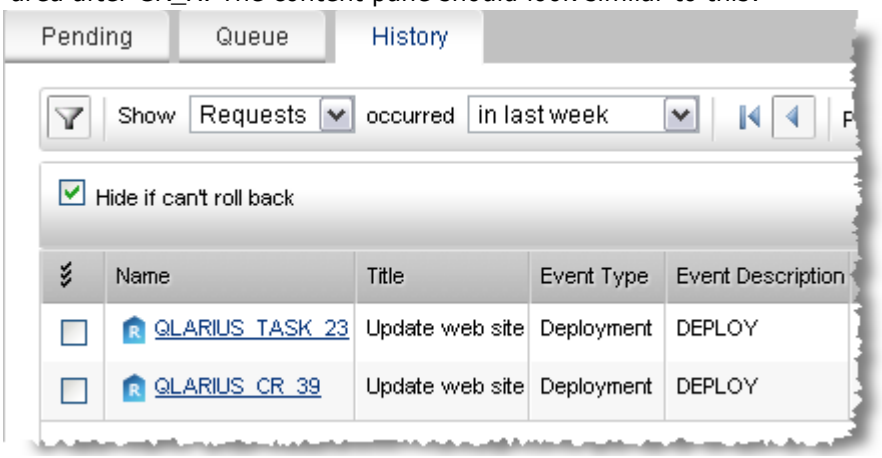
5 Make a note of the ID of the enhancement request that was raised by Rita in scenario 1 and used to deploy the changes. For the purposes of this scenario we will call this CR_X.

6 In the Items view, in the navigation pane on the Dirs tab, expand *Qlarius Underwriter* and select the *website* folder. In the content pane select the filter All Revisions.

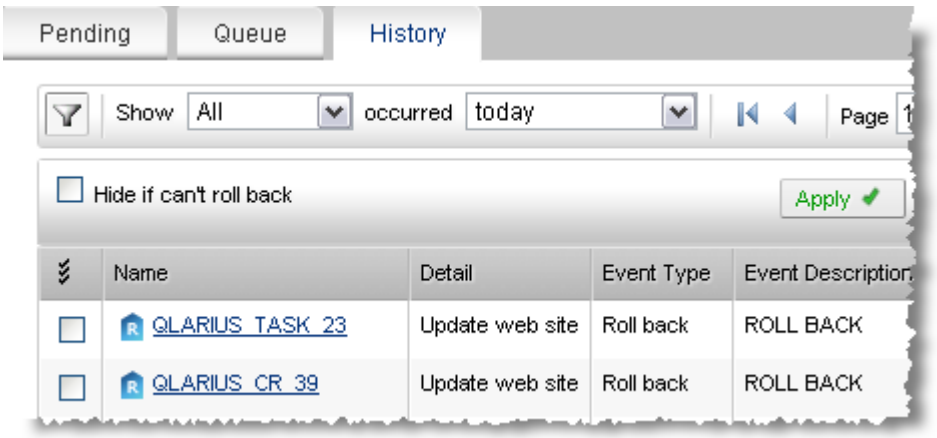
7 Make a note of the current and previous revisions of *main.css*.

8 Log out of the web client.

Running this Scenario

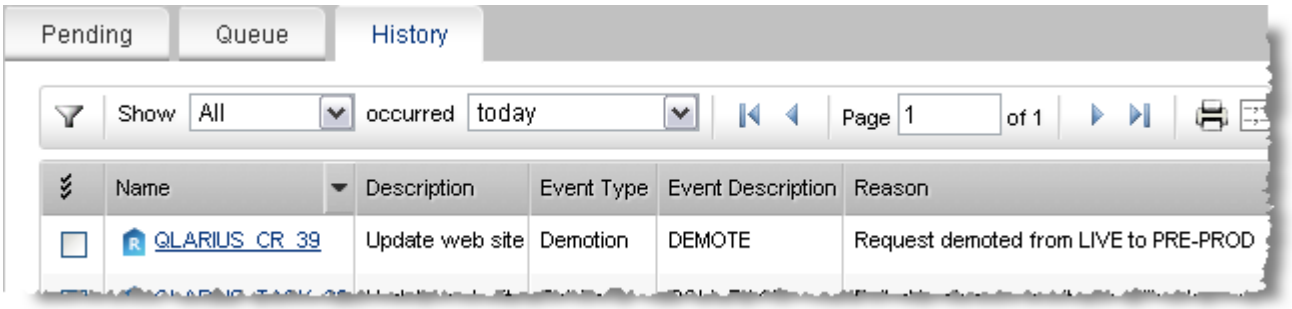
Action	Procedure
Rita, the release manager, investigates the problem with the corporate web site of Qlarius Health Insurance and discovers a small defect in *main.css,* which is related to CR_X. Rita decides that the best solution is to immediately rollback to the previous version of *main.css*.	
The release manager checks the deployment history	Rita, the release manager, checks the deployment history to see if any requests were deployed to the LIVE deployment area after CR_X. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** Select the Deployment view. **4** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **5** Select the History tab and in the navigation pane select the LIVE stage node. **6** Expand the LIVE stage node and select the deployment area LCL_LIVE_JMAIN_AREA01. **7** In the content pane use filters to only display requests that can be rolled back: **a** From the Show list select Requests. **b** From the Occurred list select the period when you performed scenario 1 (Today, in the last week, etc). **c** Click the Filter button (located to the left of the Show list). **d** Select the option Hide if can't roll back. **8** In the content pane check the history to see if any requests were deployed to the area after CR_X. The content pane should look similar to this:  • QLARIUS_CR_X is the parent enhancement request. • QLARIUS_TASK_X is the child task that was primed from CR_X. • No other requests were deployed after CR_X.

(Sheet 1 of 15)

Action	Procedure
The release manager rolls back the request from the LIVE deployment area	Rita rolls back CR_X from the LIVE deployment area. **1** On the History tab select the request QLARIUS_CR_X. **2** On the toolbar click Roll Back. The Roll Back Area Versions dialog box appears. **3** To roll back the request immediately, check that the option Perform roll back of area versions is set to 'as soon as possible'. **4** In the Reason for roll back field enter: *Defect in main.css* **5** In the Select area versions for rolling back operations table check that LCL_LIVE_JMAIN_AREA01 is selected. **6** Click Next. **7** A summary of the roll back activity and command that will be performed is displayed. **8** Click Finish. **9** Unselect Hide if can't roll back. **10** Check if the roll back succeeded. The content pane should look similar to this:  The parent request and child task have been rolled back.
The release manager checks that the previous revision of the item was redeployed	Rita checks that the rollback successfully redeployed the previous revision of *main.css* to the LIVE deployment area. **1** In the My Current Project view expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JMAIN_AREA01* > *Qlarius Underwriter* > *website* **2** In the content pane verify that the previous revision of *main.css* was deployed.

(Sheet 2 of 15)

Action	Procedure
The release manager demotes the request to the PRE-PROD stage.	Rita demotes request CR_X to the PRE-PROD stage to indicate that it is no longer live. **1** In the Deployment view, on the History tab, select the LCL_LIVE_JMAIN_AREA01 area in the navigation pane. **2** In the content pane select CR_X and on the toolbar click Demote. The Demote dialog box appears. **3** Check that the option Demote child requests is selected. **4** Check that the To Stage is set to PRE-PROD. **5** Click Next. **6** To deploy the requests now, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field select the LCL_PP_JMAIN_AREA01 deployment area. **8** Click Next. A summary of the demote and deployment activity and command that will be performed is displayed. **9** Click Finish. **10** In the navigation pane select the LIVE stage node. **11** In the content pane verify that request CR_X was successfully demoted from LIVE to PRE-PROD. The content pane should look similar to this:



(Sheet 3 of 15)

Action	Procedure
The release manager raises a change request to track the defect	Rita raises a change request to fix and track the defect. **1** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **2** In the Title field enter a title for the request, for example: *Fix main.css.* **3** In the Detailed description field enter a description. **4** On the Attributes tab, from the Severity/Priority list select Really Urgent. **5** Click Submit and click Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.
The release manager delegates the request to the team lead	Rita delegates the request to the development team lead, Ted, whose team is responsible for maintaining the web site. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.
The release manager actions the request to its next state	Rita actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The request is removed from Rita's request inbox. **4** Log out of the web client.

Action	Procedure
The development team lead primes a child task from the request	Ted reads the email, views the request in his Request inbox, and primes a child task from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Request view select the Request inbox and then the request that was raised by Rita. **4** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **5** (Optional) Update the detailed description. **6** Click Submit and click Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised.
The development team lead delegates the task to a web developer	Ted delegates the task to a web developer, Wendy. **1** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Wendy and click Add. The wizard closes automatically. Dimensions CM sends an email to Wendy notifying her that a task has been added to her Request inbox.
The development team lead actions the task to its next state	Ted actions the task to its next state, UNDER WORK. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The child task is removed from Ted's request inbox. **4** Log out of the web client.

Action	Procedure
The web developer updates their work area from the stream	Wendy reads the email, checks her Request inbox, and updates her work area from the stream to make sure she has the latest version of *main.css*. **1** Log into the web client as Wendy. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** Check that Wendy's work area is correct (see the pre-requisites at the start of scenario 1 on page 93). **4** In the Items view, on the Dirs tab of the navigation pane, expand Qlarius Underwriter and select *website*. **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next. **7** Click Finish and then Close. Wendy's work area is updated.
The web developer modifies the item	In Wendy's local work area on your machine edit *main.css*. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the file.
The web developer delivers the item and relates it to the task	Wendy delivers the modified file to the stream and relates it to the task. **1** In the Items view, on the Dirs tab of the navigation pane, select *website*. **2** On the toolbar click Deliver. The Deliver to Stream wizard appears. **3** Check that the Modifications check box is selected. **4** Click Next. **5** Verify that *main.css* is selected and click Next. **6** In the Relate to request(s) field click Select. The Select Request dialog box appears. **7** From the Product name list select QLARIUS. **8** From the Type name list select TASK. **9** Click Next. **10** Select the task that is delegated to Wendy and click Finish. **11** In the Deliver to Stream wizard click Finish and then Close. **12** Make a note of the latest revision of *main.css* (see the content pane).

(Sheet 6 of 15)

Action	Procedure
The developer verifies that the item was automatically deployed to the DEV areas	Deploy by default is enabled for the DEV area so when Wendy delivered the item it was automatically deployed. She checks that the item was successfully deployed. **1** Select the Deployment view and check that only the current stream is displayed. **2** Select the History tab and in the navigation pane select the DEV stage node. **3** In the content pane verify that *main.css* was successfully deployed to the DEV deployment area LCL_DEV_JMAIN_AREA01. The Event Result column should display *Succeeded*.
The web developer delegates the task to the team lead for peer review	Wendy delegates the child task to Ted, her team lead, for peer review. **1** In the Requests view select the Request inbox. **2** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.
The web developer actions the task to its next state	Wendy actions the child task to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Wendy's Request inbox. **7** Log out of the web client.

Action	Procedure
The team lead does a peer review and actions the task to its final state	Ted has read his email, seen the task in his Request inbox, done a peer review of the file that Wendy modified, and is satisfied with the changes that she made. He actions the task to its final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox. **3** Select the child task and on the toolbar select Action. The Action wizard appears. **4** Check that the To next state field is set to CLOSED. **5** Click Finish and click OK. The task is removed from Ted's request inbox.
The team lead promotes and deploys the request and task to the SIT stage	To perform system integration testing, Ted promotes and deploys the parent request with the task to the SIT stage and its associated deployment area. Deploy by default is enabled for the SIT area. **1** Select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that SIT is selected. **5** Click Next. **6** To deploy now, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field check that the LCL_SIT_JMAIN_AREA01 deployment area is selected. **8** Click Next. A summary of the promotion and deployment activities and command that will be performed is displayed. **9** Click Finish.

(Sheet 8 of 15)

Action	Procedure
The team lead verifies that the promotion and deployment were successful	Ted verifies that the promotion and deployment operations were executed successfully. **1** Select the Deployment view and check that only the current stream is displayed. **2** Check that the History tab is selected. **3** In the navigation pane select the SIT stage node. **4** In the content pane verify that the request was promoted successfully from DEV to SIT. The Event Result column should display *Succeeded*. **5** In the navigation pane expand the SIT stage node and select the LCL_SIT_JMAIN_AREA01 deployment area. **6** In the content pane verify that the request and task were successfully deployed to the SIT deployment area.
Ted performs system integration testing.	
The team lead promotes the request and task to the QA stage	System integration testing has been completed successfully so Ted promotes the request and task to the QA stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the QA deployment area. **1** On the History tab select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that QA is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the navigation pane select the QA stage node. **9** In the content pane verify that the request was promoted successfully from SIT to QA.

Action	Procedure
The team lead actions the request to its next state	Ted actions the parent request to its next lifecycle state, IN TEST, so that the QA team can perform testing. **1** On the History tab select the request. **2** On the toolbar click Action. The Action wizard appears. **3** In the New State section check that the To next state field is set to IN TEST. **4** Click Next. **5** In the Details of solution given field enter: *Fixed defect in main.css updated* **6** Click Finish. Dimensions CM sends an email to Tao, the QA manager, notifying her that a task has been added to his Request inbox **7** Log out of the web client.

(Sheet 10 of 15)

Action	Procedure
The QA manager deploys the request and task to the QA deployment area	Tao, the QA manager, reads the email and checks the Pending tab for the QA stage on the Deployment view. Tao sees that the request is ready to be deployed to QA. **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Deployment view check that only the current stream is displayed. **4** Select the Pending tab. **5** In the navigation pane select the QA stage node. **6** In the content pane, from the Show list select Requests. **7** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **8** Check that the option Deploy child requests is selected. **9** Check that the Deploy Stage is set to QA. **10** Click Next. **11** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **12** In the Areas(s) for deployment field check that the LCL_QA_JMAIN_AREA01 deployment area is selected. **13** Click Next. A summary of the deployment activity and command that will be performed is displayed. **14** Click Finish. **15** Select the History tab. **16** In the navigation pane expand the QA stage node and select the LCL_QA_JMAIN_AREA01 deployment area. **17** In the content pane verify that the request and child task were successfully deployed to the QA area.
The QA team performs their tests.	

(Sheet 11 of 15)

Action	Procedure
The QA manager promotes the request and task to the PRE-PROD stage	QA testing has been complete successfully so Tao promotes the request and task to the PRE-PROD stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the PRE-PROD deployment area. **1** On the History tab select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that PRE-PROD is selected. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the navigation pane select the PRE-PROD stage node. **9** In the content pane verify that the request was promoted successfully from QA to PRE-PROD.
The QA manager actions the request to its final lifecycle state	Tao closes the request that is tracking the defect. **1** On History tab select the request. **2** On the toolbar click Action. The Action wizard appears. **3** Check that the To next state field is set to CLOSED. **4** Click Finish and click OK. **5** Log out of the web client.

(Sheet 12 of 15)

Action	Procedure
The release manager deploys the request and task to the PRE-PROD deployment area	Rita, the release manager, checks the Pending tab for the PRE-PROD stage on the Deployment view. Rita sees that the request is ready to be deployed to PRE-PROD. **1** Log into the web client as Rita. **2** In the Deployment view check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane select the PRE-PROD stage node. **5** In the content pane, from the Show list select Requests. **6** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to PRE-PROD. **9** Click Next. **10** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the LCL_PP_JMAIN_AREA01 deployment area is selected. **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish and click Close. **14** Select the History tab. **15** In the navigation pane, in the PRE-PROD stage node, select the LCL_PP_JMAIN_AREA01 deployment area. **16** In the content pane verify that the request and child task were successfully deployed to the PRE-PROD area.
The release team performs their tests.	

(Sheet 13 of 15)

Action	Procedure
The release manager promotes the request and task to the LIVE stage	Rita promotes the request and task to the LIVE stage. Deploy by Default is not enabled for the LIVE deployment area. **1** On the History tab, with the PRE-PROD stage node selected in the navigation pane, select the request in the content pane. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that LIVE is selected. **5** Click Next. Rita has the privilege to deploy at the same time as the promotion but chooses not to select any deployment areas. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish. **8** In the navigation pane select the LIVE stage node. **9** In the content pane verify that the request was promoted successfully from PRE-PROD to LIVE.
The release manager deploys the request and task to the LIVE deployment area	Let's assume it is now the regular nightly maintenance period when the LIVE deployment area is offline. Rita deploys the request and task to the LIVE deployment area. **1** On the Pending tab select the request and on the toolbar click Deploy. The Deploy wizard appears. **2** Check that the option Deploy child requests is selected. **3** Check that the Deploy Stage is set to LIVE. **4** Click Next. **5** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **6** In the Areas(s) for deployment field check that the LCL_LIVE_JMAIN_AREA01 deployment area is selected. **7** Click Next. A summary of the deployment activity and command that will be performed is displayed. **8** Click Finish and click Close.

Action	Procedure
The release manager verifies that the deployment operation was successful	Rita verifies that the deployment operation was successful. **1** Select the History tab. **2** In the navigation pane expand the LIVE stage node and select the LCL_LIVE_JMAIN_AREA01 deployment area. **3** In the content pane verify that the request and task were successfully deployed to the LIVE deployment area. **Tip**: In the content pane you can see the request that was raised to fix the defect and request CR_X, which you rolled back at the start of this scenario.
The release manager verifies that the fix was deployed	Rita verifies that the latest revision of *main.css* was deployed to the LIVE area. **1** In the My Current Project view, in the navigation pane, expand *Deployment Areas > LIVE Stage > LCL_LIVE_JMAIN_AREA01 > Qlarius Underwriter > website* **2** In the content pane verify that the latest revision of *main.css* was deployed. This revision has replaced the one that was redeployed by the rollback operation at the start of this scenario.
End of scenario	

(Sheet 15 of 15)

Scenario Privileges

The tables below list the promotion and deployment privileges required by each user in the scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT
	QA Manager	QA
	Release Manager	PRE-PROD

Deployment privilege	Privilege owner	Required for these areas
The DEV and SIT areas are deploy by default areas and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	QA Manager	LCL_QA_JMAIN_AREA01
	Release Manager	LCL_PP_JMAIN_AREA01 LCL_LIVE_JMAIN_AREA01

Deployment privilege	Privilege owner	Required for these areas
REQUEST_ROLLBACK_ANYSTAGE ITEM_ROLLBACK_ANYSTAGE **Note**: These privileges are only used for rolling back area versions and will change in a future release.	Release Manager	LCL_PP_JMAIN_AREA01 LCL_LIVE_JMAIN_AREA01
REQUEST_DEMOTE_ANYSTAGE ITEM_DEMOTE_ANYSTAGE	Release Manager	LCL_PP_JMAIN_AREA01 LCL_LIVE_JMAIN_AREA01

Scenario 6: Deploying a Fix Forward Solution using a Request

Scenario Objective

A defect at the QA stage is preventing testing on the Qlarius web application. The objective of this scenario is to prepare a fix and deploy if forward over the part of the application that is not working.

Scenario Overview

- The QA manager raises a change request to track the defect.
- The development team lead primes a task from the request.
- A developer fixes the defect, delivers the fix and relates it to the task, builds the task, and captures the build outputs in Dimensions CM.
- The team lead promotes and deploys the request and task to the SIT stage and deployment area, and then promotes them to the QA stage.
- The QA manager deploys the request and task to the QA deployment area.

Scenario Information

- The following stream is used: QLARIUS:JAVA_BRANCHA_STR
- There is a division of responsibilities between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- A build is required at the DEV stage.
- This scenario uses the following deployment areas:

Stage	Deployment area	Deploy by Default enabled for area?
DEV	LCL_DEV_JBRNCHA_AREA03	Yes
SIT	LCL_SIT_JBRNCHA_AREA03	Yes
QA	LCL_QA_JBRNCHA_AREA03	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 191.

Pre-Requisites

1 To run this scenario you must have successfully completed "Scenario 3: Deploying Requests to Multiple Deployment Areas" on page 123.

2 Log into the web client as any user.

3 Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR

4 Browse the QA web application deployment area: in the My Current Project view expand *Deployment Areas* > *QA Stage* > *LCL_QA_JBRNCHA_AREA03* > *Qlarius Underwri*ter.

5 In the content pane make a note of the revision of *AutoQuote.jar* that is currently deployed in the QA deployment area.

6 Log out of the web client.

Running this Scenario

Action	Procedure
Tao, the QA manager, does some research and finds that the problem is in a single file, *AutoQuote.jar*, in the LCL_QA_JBRNCHA_AREA03 web application deployment area. Tao decides that the quickest solution is to prepare a fix and deploy it forward over the file that is causing the problem.	
The QA manager raises a change request to track the defect	Tao raises a change request to fix and track the defect. **1** Log into the web client as Tao. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR. **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter a title for the request, for example: *Fix Autoquote.java.* **5** In the Detailed description field enter a description. **6** On the Attributes tab, from the Severity/Priority list select a priority. **7** Click Submit and click Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.

(Sheet 1 of 11)

Action	Procedure
The QA manager delegates the request to the team lead	Tao delegates the request to the development team lead, Ted, whose team is responsible for maintaining Qlarius. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.
The QA manager actions the request to its next state	Tao actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The request is removed from Tao's request inbox. **4** Log out of the web client.
The development team lead primes a child task from the request	Ted reads the email, views the request in his Request inbox, and primes a child task from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** In the Request view select the Request inbox and then the request that was raised by Tao. **4** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **5** (Optional) Update the detailed description. **6** Click Submit and click Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised.

(Sheet 2 of 11)

Action	Procedure
The development team lead delegates the task to a developer	Ted delegates the child task to a developer, Dinesh. **1** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Dinesh and click Add. The wizard closes automatically. Dimensions CM sends an email to Dinesh notifying him that a task has been added to his Request inbox.
The development team lead actions the task to its next state	Ted actions the task to its next state, UNDER WORK. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The child task is removed from Ted's request inbox. **4** Log out of the web client.
The developer updates their work area from the stream	Dinesh reads the email, checks his Request inbox, and updates his work area. **1** Log into the web client as Dinesh. **2** Switch to the following stream: QLARIUS:JAVA_BRANCHA_STR **3** Change Dinesh's work area to the one that you created earlier (see the pre-requisites at the start of scenario 3). **4** In the Items view, on the navigation pane select the folder that contains *Autoquote.java*: *Qlarius Underwriter* > *qlarius* > *interfaces* **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next, **7** Click Finish and then Close. Dinesh's work area is updated.
The developer fixes the defect	In Dinesh's local work area edit *Autoquote.java*. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the item.

(Sheet 3 of 11)

Action	Procedure
The developer delivers the item and relates it to the task	Dinesh delivers the fixed item to Dimensions CM and relates it to the task. **1** In the Items view, on the toolbar click Deliver. The Deliver to Stream wizard appears. **2** Check that the Modifications check box is selected. **3** Click Next. **4** Verify that *AutoQuote.java* is selected and click Next. **5** In the Relate to request(s) field click Select. The Select Request dialog box appears. **6** From the Product name list select QLARIUS. **7** From the Type name list select TASK. **8** Click Next. **9** Select the task that is delegated to Dinesh and click Finish. **10** In the Deliver to Stream wizard click Finish and then Close.
The developer verifies that the item was automatically deployed to the DEV area	Deploy by default is enabled for the DEV area so when Dinesh delivered the item it was automatically deployed. He checks that the item was successfully deployed. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Select the History tab and in the navigation pane select the DEV stage node. **4** In the content pane verify that the *AutoQuote.java* was successfully deployed to the LCL_DEV_JBRNCHA_AREA03 deployment area.

(Sheet 4 of 11)

Action	Procedure
The developer builds the task and captures the outputs	Dinesh builds the task in the DEV web application deployment area and captures the build outputs in Dimensions CM against the task. **1** On the Requests view select the Request inbox. **2** Select the task, on the toolbar click Build, and select Build. The Run Build wizard appears. **3** In the Build Configuration field accept the default configuration. **4** From the Build Stage list select DEV. **5** From the Build Area list select LCL_DEV_JBRNCHA_AREA03. **6** Click Next. **7** Select the option Check in build outputs automatically. This will check the build outputs into Dimensions CM. **8** To specify the request that the build outputs will be related to when they are checked into Dimensions CM, in the Specify the request(s) field click Select. The Select Request wizard appears. **9** Do the following: **a** From the Product name list select QLARIUS. **b** From the Type name list select TASK. **c** Click Next. **d** Select the task that is delegated to Dinesh. **e** Click Finish. **10** In the Run Build wizard click Next. **11** Accept the default build option selections (none) and click Next. **12** Accept the default target selection options. **13** In the target list select *Jar Files*. **14** Click Next. A summary of the build command that will be executed is displayed. **15** Click Finish. **16** (Optional) To monitor the progress of the build click the Job <ID number> link. **17** Click Close.

(Sheet 5 of 11)

Action	Procedure
The developer verifies that the build was successful	Dinesh verifies that the build was successful and that the outputs were deployed to the DEV web application deployment area. **1** To verify that the build was successful, in the Deployment view select the History tab. **2** In the navigation pane expand the DEV stage node and select LCL_DEV_JBRNCHA_AREA03. **3** In the content pane verify that the Event Result column displays *Succeeded* for the following objects: • Autoquote.jar (the Event Type is Collect) • QLARIUS:JAVA_BRANCHA_STR (the Event Type is Build) **4** Open *Autoquote.jar*, make a note of the item revision, and click Cancel. **5** To browse the DEV web application deployment area, in the My Current Project view expand *Deployment Areas > DEV Stage > LCL_DEV_JBRNCHA_AREA03* **6** Select the folder *Qlarius Underwriter*. **7** In the content pane verify that the latest revision of *AutoQuote.jar* is deployed. This should be later than the revision that you made a note of at the start of this scenario.
The developer delegates the task to the team lead for peer review	Dinesh delegates the task to Ted for peer review. **1** In the Requests view select the Request inbox. **2** Select the task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.

Action	Procedure
The developer actions the task to its next state	Dinesh actions the task to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Dinesh's request inbox. **7** Log out of the web client.
The team lead does a peer review and actions the task to its final state	Ted has read his email, seen the task in his Request inbox, done a peer review of the defect that Dinesh fixed, and is satisfied with the changes that he made. He actions the task to its final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox, select the task, and on the toolbar click Action. The Action wizard appears. **3** Check that the To next state field is set to CLOSED. **4** Click Finish and click OK. The task is removed from Ted's Request inbox.

Action	Procedure
The team lead promotes and deploys the request and the task to the SIT stage and area	To perform system integration testing, Ted promotes and deploys the request and task to the SIT stage and web application deployment area. **1** In the Request view, in the Request inbox, select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that SIT is selected. **5** Click Next. **6** To deploy immediately, check that the option Perform deployments is set to 'as soon as possible'. **7** In the Areas(s) for deployment field select the SIT LCL_SIT_JBRNCHA_AREA03 deployment area (you might need to deselect LCL_SIT_JBRNCHA_AREA01). **8** Click Next. A summary of the deployment activity and command that will be performed is displayed. **9** Click Finish.
The team lead verifies that promotion and deployment operations were successful	Ted verifies that promotion and deployment operations were successful and that the modified file was deployed to the SIT web application deployment area. **1** Select the Deployment view and check that only the current stream is displayed. **2** Select the History tab and in the navigation pane select the SIT stage node. **3** In the content pane verify that the request was promoted successfully from DEV to SIT. **4** In the navigation pane expand the SIT stage node and select the LCL_SIT_JBRNCHA_AREA03 deployment area. **5** In the content pane verify that the request and task were deployed successfully to the SIT web application deployment area. **6** To browse the SIT web application deployment area, in the My Current Project view expand *Deployment Areas > SIT Stage > LCL_SIT_JBRNCHA_AREA03.* **7** Select the folder *Qlarius Underwri*ter. **8** In the content pane verify that latest revision of *AutoQuote.jar* is deployed.
Ted performs system integration testing.	

(Sheet 8 of 11)

Action	Procedure
The team lead promotes the request and task to the QA stage	System integration testing has been completed successfully so Ted promotes the request and task to the QA stage. Deploy by Default is not enabled so the request and task cannot be automatically deployed to the QA deployment area. **1** In the Requests view, in the Request inbox, select the request and on the toolbar click Promote. The Promote wizard appears. **2** Check that the option Promote child requests is selected. **3** In the Next stage field check that QA is selected. **4** Click Next. Deploy by Default is not enabled so no deploy options are available. **5** Click Next. A summary of the promotion activity and command that will be performed is displayed. **6** Click Finish. Dimensions CM sends an email to the QA manager, Tao, notifying her that a promotion has been performed. **7** In the Deployment view select the History tab and in the navigation pane select the QA stage node. **8** In the content pane verify that the request was promoted successfully from SIT to QA.
The team lead actions the request to its next state	Ted actions the request to its next lifecycle state, IN TEST, so that QA can test the application. **1** On the History tab select the request and on the toolbar click Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to IN TEST. **3** Click Next. **4** In the Details of solution given field enter a solution. **5** Click Finish. Note: The task is removed from Ted's request inbox. **6** Log out of the web client.

(Sheet 9 of 11)

Action	Procedure
The QA manager deploys the request and task to the QA deployment area	Tao reads the email, checks her Request inbox, and deploys the request and task to the QA web application deployment area. **1** Log into the web client as Tao. **2** Select the Deployment view and check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane expand the QA stage node and select the LCL_QA_JBRNCHA_AREA03 deployment area. **5** In the content pane, from the Show list select Requests. **6** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to QA. **9** Click Next. **10** To deploy the request and child tasks immediately, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the LCL_QA_JBRNCHA_AREA03 deployment area is selected. **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish.
The QA manager verifies that promotion and deployment operations were successful	Tao verifies that promotion and deployment operations were successful and that the modified file was deployed to the QA web application deployment area. **1** Select the History tab. **2** In the content pane verify that the request and task were deployed successfully. **3** To browse the QA web application deployment area, in the My Current Project view expand *Deployment Areas > QA Stage > LCL_QA_JBRNCHA_AREA03.* **4** Select the folder *Qlarius Underwri*ter. **5** In the content pane verify that latest revision of *AutoQuote.jar* is deployed.
The QA team performs their tests.	

(Sheet 10 of 11)

Action	Procedure
The QA manager actions the request to its final lifecycle state.	QA testing has been completed successfully so Tao closes the enhancement request that she raised against the defect. **1** In the Request view select the request and on the toolbar click Action. The Action wizard appears. **2** Check that the To next state field is set to CLOSED. **3** Click Finish and click OK. The request is removed from Tao's request inbox.
End of scenario	

(Sheet 11 of 11)

Scenario Privileges

The tables below list the promotion and deployment privileges required by each user in the scenario.

Promotion privilege	Privilege owner	Required at these stages
REQUEST_PROMOTE_NEXTSTAGE ITEM_PROMOTE_NEXTSTAGE	Team lead	DEV SIT

Deployment privilege	Privilege owner	Required for these areas
The DEV and SIT areas are deploy by default areas and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	QA Manager	LCL_QA_JBRNCHA_AREA02

Scenario 7: Deploying an Emergency Fix

Scenario Objective

After an update to the corporate web site of Qlarius Health Insurance a defect is found that prevents the site from being used by the customers. Rolling back to the previous version is not a solution as changes were introduced that the company does not want to lose. The objective of this scenario is to apply a quick solution by making an emergency fix.

Scenario Overview

- The release manager raises a change request to track the defect.
- The development team lead primes a task from the request.
- A web developer fixes and delivers the defect, and relates it to the task.
- The team lead promotes the request and task to the PRE-PROD stage.
- The release manager deploys the request and task to the PRE-PROD deployment area, and then promotes and deploys them to the LIVE stage and production environment.

Scenario Information

- The following stream is used: QLARIUS:MAINLINE_JAVA_STR
- There is a division of responsibilities between the employees at the following stage transitions:
  - SIT to QA
  - QA to PRE-PROD
- No build is required at any stage as only a text file is changed.
- This scenario uses the following deployment areas:

Stage	Deployment area	Deploy by Default enabled for area?
DEV	LCL_DEV_JMAIN_AREA01	Yes
PRE-PROD	LCL_PP_JMAIN_AREA01	No
LIVE	LCL_LIVE_JMAIN_AREA01	No

- For a list of the promotion and deployment privileges required by the users see "Scenario Privileges" on page 202.

Pre-Requisites

1. Log into the web client as any user.
2. Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR
3. Browse the LIVE web application deployment area: in the My Current Project view expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JMAIN_AREA01* > *Qlarius Underwriter* > *website*.
4. In the content pane make a note of the revision of *index.html* that is currently deployed in this area.
5. Log out of the web client.

Running this Scenario

Action	Procedure
Rita, the release manager, investigates the problem and discovers a defect in *index.html*. Rita decides that the best solution is to deploy an emergency fix.	
The release manager raises a change request	Rita raises a change request to fix and track the defect. **1** Log into the Dimensions CM web client as Rita. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR. **3** In the Request view, on the toolbar click New and select CR. The New Request dialog box appears. **4** In the Title field enter a title for the request, for example: *Fix index.html.* **5** In the Detailed description field enter a description. **6** On the Attributes tab, from the Severity/Priority list select Really Urgent. **7** Click Submit and click Close. The new request is added to Rita's request inbox with the following ID: QLARIUS_CR_*n* By default the request is at the DEV stage when it is raised.

(Sheet 1 of 9)

Action	Procedure
The release manager delegates the request to the team lead	Rita delegates the request to the development team lead, Ted, whose team is responsible for maintaining the web site. **1** Select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a request has been added to his Request inbox.
The release manager actions the request to its next state	Rita actions the request to its next state, UNDER WORK. **1** Select the request and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The request is removed from Rita's request inbox. **4** Log out of the web client.
The development team lead primes a task from the request	Ted reads the email, views the request in his Request inbox, and primes a child task from the request. **1** Log into the web client as Ted. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** In the Request view select the Request inbox and then the request that was raised by Rita. **4** On the toolbar click Prime and select Task. The Prime Request dialog box appears. **5** (Optional) Update the detailed description. **6** Click Submit and click Close. The new child task is added to Ted's request inbox with the following ID: QLARIUS_TASK_*n* By default the child task is at the DEV stage when it is raised.

(Sheet 2 of 9)

Action	Procedure
The development team lead delegates the task to a web developer	Ted delegates the task to a web developer, Wendy. **1** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select IMPLEMENTOR and click Next. **4** In the 'Candidate users authorized for role assignment' list select Wendy and click Add. The wizard closes automatically. Dimensions CM sends an email to Wendy notifying her that a task has been added to her Request inbox.
The development team lead actions the task to its next state	Ted actions the task to its next state, UNDER WORK. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** Check that the To next state field is set to UNDER WORK. **3** Click Finish and click OK. The child task is removed from Ted's request inbox. **4** Log out of the web client.
The web developer updates their work area from the stream	Wendy reads the email, checks her Request inbox, and updates her work area from the stream to make sure she has the latest revision of *index.html*. **1** Log into the web client as Wendy. **2** Switch to the following stream: QLARIUS:MAINLINE_JAVA_STR **3** Check that Wendy's work area is correct (see the pre-requisites at the start of scenario 1 on page 93). **4** In the Items view, on the Dirs tab of the navigation pane, expand Qlarius Underwriter and select *website*. **5** On the toolbar click Update. The Update from Stream wizard appears. **6** Click Next. **7** Click Finish and then Close. Wendy's work area is updated.
The web developer modifies the item	In Wendy's local work area on your machine edit *index.html*. For the purpose of this scenario make a minor edit, for example, add a comment to the top of the file.

(Sheet 3 of 9)

Action	Procedure
The web developer delivers the item and relates it to the task	Wendy delivers the modified file to the stream and relates it to the child task. **1** In the Items view, on the Dirs tab of the navigation pane, select *website*. **2** On the toolbar click Deliver. The Deliver to Stream wizard appears. **3** Check that the Modifications check box is selected. **4** Click Next. **5** Verify that *index.html* is selected and click Next. **6** In the Relate to request(s) field click Select. The Select Request dialog box appears. **7** From the Product name list select QLARIUS. **8** From the Type name list select TASK. **9** Click Next. **10** Select the task that is delegated to Wendy and click Finish. **11** In the Deliver to Stream wizard click Finish and then Close. **12** Make a note of the latest revision of *index.html* (see the content pane).
The web developer delegates the task to the team lead for peer review	Wendy delegates the task to Ted, her team lead, for peer review. **1** In the Requests view select the Request inbox. **2** Select the child task and on the toolbar click Delegate. The Delegate wizard appears. **3** Check that the Capability option is set to Secondary. **4** From the Role to Delegate list select REVIEWER and click Next. **5** In the 'Candidate users authorized for role assignment' list select Ted and click Add. The wizard closes automatically. Dimensions CM sends an email to Ted notifying him that a task has been added to his Request inbox.

(Sheet 4 of 9)

Action	Procedure
The web developer actions the task to its next state	Wendy actions the task to its next state, PEER REVIEW. **1** Select the child task and on the toolbar select Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to PEER REVIEW. **3** Click Next. **4** In the Actual completed date field enter a date. **5** In the Actual development effort (hours) field enter a number. **6** Click Finish. The task is removed from Wendy's Request inbox. **7** Log out of the web client.
The team lead does a peer review and actions the task to its final state	Ted has read his email, seen the task in his Request inbox, done a peer review of the file that Wendy modified, and is satisfied with the changes that she made. He actions the task to its final state, CLOSED. **1** Log into the web client as Ted. **2** In the Requests view select the Request inbox. **3** Select the child task and on the toolbar select Action. The Action wizard appears. **4** Check that the To next state field is set to CLOSED. **5** Click Finish and click OK. The task is removed from Ted's request inbox.

(Sheet 5 of 9)

Action	Procedure
The team lead promotes the request and task straight to the PRE-PROD stage	Because this is an emergency fix that is required urgently, Ted promotes the request and task straight from DEV to PRE-PROD bypassing the intermediate stages (SIT and QA). Deploy by Default is not enabled so the request and task cannot be automatically deployed to the PRE-PROD deployment area. **1** Select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** From the Next stage field select PRE-PROD. **5** Click Next. Deploy by Default is not enabled so no deploy options are available. **6** Click Next. A summary of the promotion activity and command that will be performed is displayed. **7** Click Finish.
The team lead verifies that promotion was successful	Ted verifies that promotion to PRE-PROD was successful. **1** Select the Deployment view. **2** To only display information for the current stream do the following: **a** In the navigation pane click the filter button in the top right corner. **b** Select Show Current Stream. **3** Select the History tab. **4** In the navigation pane select the PRE-PROD stage node. **5** In the content pane verify that the request was promoted successfully from DEV to PRE-PROD.
The team lead delegates the request to a QA engineer	Ted delegates the request to Tony, a QA engineer, for testing. **1** In the Requests view select the request and on the toolbar click Delegate. The Delegate wizard appears. **2** Check that the Capability option is set to Secondary. **3** From the Role to Delegate list select QA ENGINEER and click Next. **4** In the 'Candidate users authorized for role assignment' list select Tony and click Add. The wizard closes automatically. Dimensions CM sends an email to Tony notifying him that a task has been added to his Request inbox.

(Sheet 6 of 9)

Action	Procedure
The team lead actions the request to its next state	Ted actions the request to its next lifecycle state, IN TEST, so that the Tony can perform testing. **1** Select the request and on the toolbar click Action. The Action wizard appears. **2** In the New State section check that the To next state field is set to IN TEST. **3** Click Next. **4** In the Details of solution given field enter: *Fixed index.html* **5** Click Finish. The task is removed from Ted's Request inbox. Dimensions CM sends an email to Tony notifying him that a task has been added to his Request inbox. **6** Log out of the web client.
Let's assume the following: ■ Tony has tested the web site to make sure that the latest revision of *index.html* fixes the problem. ■ Tao, the QA manager, has closed the request.	

(Sheet 7 of 9)

Action	Procedure
The release manager deploys the request to the PRE-PROD deployment area	Rita, the release manager, checks the Pending tab for the PRE-PROD stage on the Deployment view. Rita sees that the request is ready to be deployed to PRE-PROD. **1** Log into the web client as Rita. **2** In the Deployment view check that only the current stream is displayed. **3** Select the Pending tab. **4** In the navigation pane select the PRE-PROD stage node. **5** In the content pane, from the Show list select Requests. **6** Select the request and on the toolbar click Deploy. The Deploy wizard appears. **7** Check that the option Deploy child requests is selected. **8** Check that the Deploy Stage is set to PRE-PROD. **9** Click Next. **10** To deploy the request and child task now, check that the option Perform deployments is set to 'as soon as possible'. **11** In the Areas(s) for deployment field check that the LCL_PP_JMAIN_AREA01 deployment area is selected. **12** Click Next. A summary of the deployment activity and command that will be performed is displayed. **13** Click Finish and click Close. **14** Select the History tab. **15** In the navigation pane, in the PRE-PROD stage node, select the LCL_PP_JMAIN_AREA01 deployment area. **16** In the content pane verify that the request and child task were successfully deployed to the PRE-PROD area.

(Sheet 8 of 9)

Action	Procedure
The release manager promotes and deploys the request and task to the LIVE stage and deployment area	Because this is an emergency fix Rita immediately promotes and deploys the request and task to the LIVE stage and deployment area. **1** On the History tab select the request. **2** On the toolbar click Promote. The Promote wizard appears. **3** Check that the option Promote child requests is selected. **4** In the Next stage field check that LIVE is selected. **5** Click Next. **6** To deploy the request and child tasks immediately, check that the option Perform deployments is set to 'as soon as possible'. **7** Deploy by Default is not enabled for the LIVE deployment area. However Rita has the privilege to deploy at the same time as the promotion. In the Areas(s) for deployment field select the LCL_LIVE_JMAIN_AREA01 deployment area. **8** Click Next. A summary of the promotion activity and command that will be performed is displayed. **9** Click Finish. **10** In the navigation pane select the LIVE stage node. **11** In the content pane verify that the request was promoted successfully from PRE-PROD to LIVE. **12** In the navigation pane select the LCL_LIVE_JMAIN_AREA01 deployment area. **13** In the content pane verify that the request and task were successfully deployed to the LIVE deployment area.
The release manager verifies that the emergency fix was deployed	Rita verifies that the latest revision of *index.html* was deployed to the LIVE area. **1** In the My Current Project view, in the navigation pane, expand *Deployment Areas* > *LIVE Stage* > *LCL_LIVE_JMAIN_AREA01* > *Qlarius Underwriter* > *website* **2** In the content pane verify that the latest revision of *index.html* was deployed. This revision has replaced the one that was deployed at the start of this scenario.
Note: After the emergency deployment is completed, the latest revision of *index.html* goes through the normal testing procedures. If it passes the tests successfully it remains in the LIVE deployment areas.	
End of scenario	

(Sheet 9 of 9)

Scenario Privileges

The tables below list the promotion and deployment privileges required by each user in the scenario.

Promotion privilege	Privilege owner
REQUEST_PROMOTE_ANYSTAGE ITEM_PROMOTE_ANYSTAGE	Team lead
	Release Manager

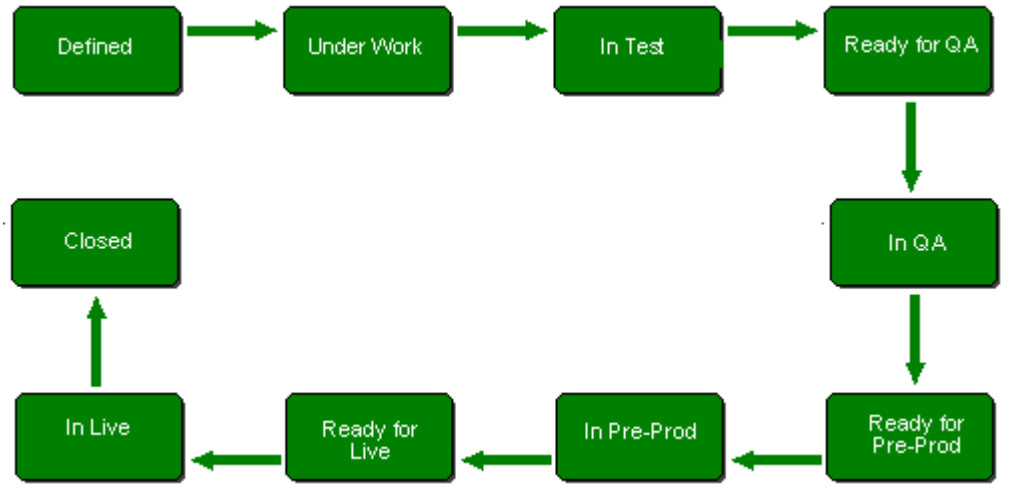
Deployment privilege	Privilege owner	Required for these areas
The DEV and LIVE areas are deploy by default areas and no deployment privileges are required.		
REQUEST_DEPLOY ITEM_DEPLOY	Release Manager	LCL_PP_JMAIN_AREA01

Scenario 8: Deploying Requests by Actioning

Scenario Introduction

An alternative way to deploy requests is to use action driven deployment. This scenario is similar to "Scenario 2: Deploying Requests to a Single Deployment Area" on page 107, the main difference is that some of the request lifecycles states are mapped to the GSL. When a request is actioned to a lifecycle state that is mapped to a stage in the GSL, it is automatically promoted to that stage. If Deploy by Default is on for that stage's deployment area, the request is automatically deployed to the area.

This scenario uses the following extended lifecycle for enhancement requests:



The following table shows request lifecycle states that are mapped to stages in the GSL.

Request lifecycle state	Mapped to GSL stage
Raised	DEV
Under Work	
In Test	SIT
Ready for QA	QA
In QA	
Ready for Pre-Prod	PRE-PROD
In Pre-Prod	
Ready for Live	
In Live	LIVE
Closed	

Scenario Overview

NOTE This scenario does not contain detailed steps.

- Rita, the release manager, does the following:
  - Raises an enhancement request to manage the change that is required to the Qlarius web application. By default the request is at the DEV stage when it is created.
  - Delegates the request to Ted, the team lead.
  - Actions the request to its next lifecycle state, *Under Work*. The request is removed from Rita's Request inbox.
- Ted does the following:
  - Does some design work to see what part of the application is affected.
  - Primes a child task from the request. By default the task is at the DEV stage when it is created.
  - Delegates the task to Dinesh, a developer.
  - Actions the task to its next lifecycle state, *Under Work*. The task is removed from Ted's Request inbox.
- Dinesh does the following:
  - Update his work area from the stream.
  - Modifies the sources.
  - Delivers the changes to the stream and relates them to the task. Deploy by Default for the DEV area is enabled so the sources are automatically deployed.
  - Delegates the task to the team lead for peer review.
  - Action the task to the next lifecycle state, *Peer Review*. The task is removed from Dinesh's Request inbox.
- Ted does the following:
  - Performs a peer review.
  - Actions the task to its final state, *Closed*. The task is removed from Ted's Request inbox.
  - Actions the request to the *In Test* state. This lifecycle state is mapped to the SIT stage therefore the request and child tasks are automatically promoted to the SIT stage. Deploy by Default for the SIT area is enabled so the request and task are automatically deployed.
  - Performs system integration tests.
  - Actions the request to the *Ready for QA* state. This state is mapped to the QA stage therefore the request and child task are automatically promoted to the QA stage. Because of the separation of duties the team lead cannot deploy the request and task to QA (Deploy by Default is not enabled for the QA deployment area).

- Tao does the following:
  - Actions the request to the *In QA* state.
  - Deploys the request and child task to the QA deployment area.
  - Delegates the request to a QA engineer who tests the request. After completing the tests the engineer delegates the request back to the QA manager.
  - Actions the request to the *Ready for Pre-Prod* state. This state is mapped to the PRE-PROD stage therefore the request and child task are automatically promoted to the PRE-PROD stage. Because of the separation of duties the QA manager cannot deploy the request and task to PRE-PROD (Deploy by Default is not enabled for the PRE-PROD deployment area).
- Rita, the release manager, does the following:
  - Actions the request to the *In Pre-Prod* state.
  - Deploys the request and the child task to the PRE-PROD deployment area.
  - Delegates the request to a release engineer who tests the request. After completing the tests the release engineer delegates the request back to the release manager.
  - Actions the request to the *Ready for Live* state.
  - During the regular maintenance period when the LIVE deployment area is offline, actions the request to the *In Live* state. This state is mapped to the LIVE stage therefore the request and child task are automatically promoted to the LIVE stage. Deploy by Default is not enabled for the LIVE stage so no deployment takes place with the promotion.
  - Deploys the request and child task to the LIVE deployment area.
  - Actions the request to its final state, *Closed*.

Chapter 8

Configuring a Deployment Environment

Configuring an Environment

Configuring a Dimensions deployment environment is typically performed by administrators in the Dimensions CM administration console. This chapter is a high level overview of the main configuration activities.

To create a deployment environment for your organization you need to configure some or all of the following:

- The Global Stage Lifecycle (GSL)
- Lifecycles for objects
- Deployment areas
- Deployment roles and privileges
- Deployment email notifications

Configuring the Global Stage Lifecycle

The GSL is the base database lifecycle that items follow through the deployment process. You can configure the Global Stage Lifecycle (GSL) as follows:

- Add or delete a transition between stages.
- Add, rename, or delete a stage.
- Assign user roles to transitions.

For details see the chapter *Area Definitions* in the *Process Configuration Guide*.

Configuring Lifecycles

A lifecycle is a set of states that defines the workflow of an object. It consists of a set of linked state transitions, each transition defining what role a user must have to move the object to the end state in the transition. You can configure lifecycles as follows:

- Create a new lifecycle for items, requests and baselines.
- Delete a lifecycle.
- Relate a lifecycle to one or more object types.
- Edit the states, transitions, roles, attribute rules, CM rules, and properties for a lifecyle.
- Map lifecycle states to the GSL.

For details see the chapter *Lifecycle Management* in the *Process Configuration Guide*.

Configuring Deployment Areas

A deployment area is a physical location on disk that contains a snapshot of the items at a particular stage in the GSL. You can configure deployment areas as follows:

- Create an area.
- Associate a stage in the GSL to an area.
- Assign an area to a project or stream.
- Set the deployment sequence for an area.
- Enable Deploy by Default for an area.
- Create and assign area scripts.
- Create and assign area filters.

For details see the chapter *Area Definitions* in the *Process Configuration Guide*.

Configuring Deployment Roles and Privileges

A privilege is a function or action that a user or group can perform, such as a privilege on the stage that you are promoting to or demoting from. There are a set of privilege rules that you can specify in the administration console for each privilege that determine which users can perform that function, and under what conditions.

For details see the chapter *Users and Roles* in the *Process Configuration Guide*.

Setting Up Deployment Email Notifications

An email notification defines an event in Dimensions CM that causes an email message to be sent to specified users or groups. For example, you can set up a notification when an item is promoted and deployed.

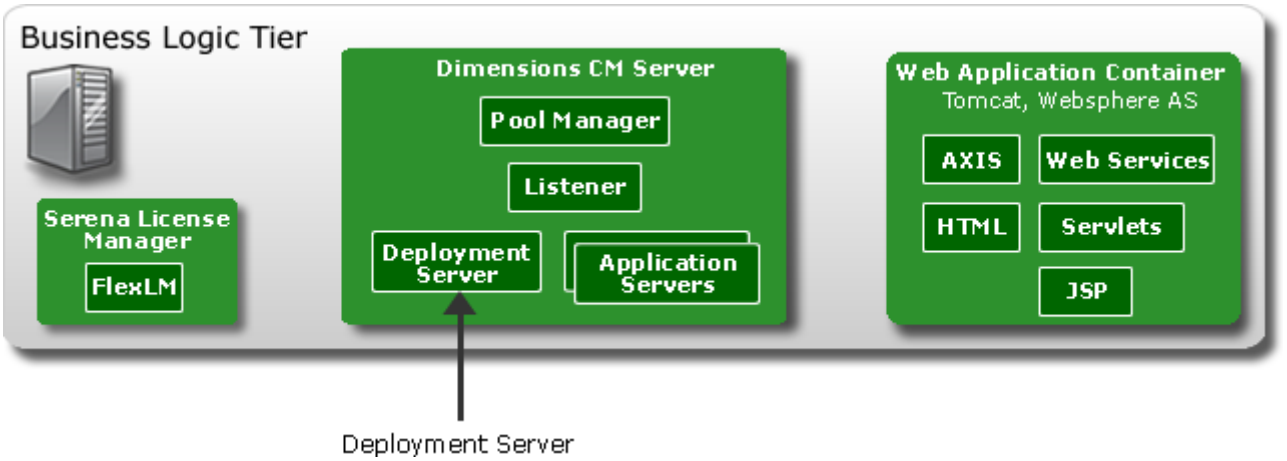
For details see the chapter *Users and Roles* in the *Process Configuration Guide*.

Chapter 9

The Dimensions Deployment Server

Introduction

You can perform Dimensions deployment operations asynchronously in the background. The asynchronous processing is executed by a Dimensions CM server component called the deployment server (dmdeploysrv.exe). The following diagram shows the deployment server in the business logic tier in the Dimensions CM architecture:



The deployment server is started by the pool manager. The deployment server then reads the configuration file (`DM_ROOT/dfs/deploy_config.dat`) and begins to monitor the deployment job queues for any new deployment jobs that need to be executed in each base database.

As users execute deployment related commands such as promote, demote, build, and audit, these commands submit new deployment jobs into the deployment job queue of the base database. A deployment job includes information about:

- The type of the operation to be performed (deploy, rollback, build, audit, clean etc).
- The deployment areas where the deployment job will be executed.
- Where appropriate, the set of Dimensions artifacts to be applied to each area, for example, the items and requests to be deployed to an area.

As new deployment jobs are added to the deployment job queue, the deployment manager automatically reads the details of the new jobs and executes them.

The deployment server uses the following rules to execute deployment jobs, which ensures that jobs are processed efficiently:

- When a deployment job is selected for execution the affected deployment areas are locked by the deployment manager so that no other deployment job can execute in the areas until the deployment job finishes. After the job is completed the areas are automatically unlocked.
- When a deployment job affects multiple deployment areas it is executed concurrently in as many deployment areas as possible, depending on:

  - The sequence numbers assigned to each area. For details about setting sequence numbers see "Setting a Deployment Sequence" on page 37.
  - The availability of system resources such as free application servers in the pool.
- Deployment jobs submitted in a single user session are executed sequentially in the order that they were submitted as soon as all the deployment areas affected by each job are available.
- Deployment jobs submitted from different user sessions are executed concurrently in the order that they were submitted as soon as all the deployment areas affected by each job are available.

The deployment server does not access the deployment areas. Instead, it uses application servers to perform any relevant database meta data queries, obtain connections to library servers on deployment area nodes, and carry out the deployment operations in each area.

Configuring a Deployment Server

A single deployment server process can monitor one or more deployment queues in multiple base databases. You configure the deployment server in `DM_ROOT/dfs/deploy_config.dat file`. By default, the configuration file is set up by the installer to monitor the default database in the listener configuration file (`DM_ROOT/dfs/listener.dat`). The configuration parameters in `deploy_config.dat` are organized in two groups:

- Global parameters that apply to all base databases.
- Base database-specific parameters.

Global Parameters

Parameter	Description	Default value
`log_dir`	Specify this parameter if you want to: ■ Enable logging. ■ Redirect the log to a file in the directory specified by this variable.	None, logging is off.
`idle_timeout`	Specifies the timeout after which the deployment server will close unused application server connections. If you set the parameter to 0 unused connections will be kept in the connection pool until the deployment server terminates.	60

Base Database Parameters

Parameter	Description	Default value
`database` `database_[n]`	Specifies the database to be monitored for new deployment jobs. You can specify any number of databases to monitor using the format `database_[n]` where: `n = 1, 2, 3, ...` You must specify at least one database otherwise the deployment server will be inactive.	None
`host` `host_[n]`	Specifies the application server host and port that is working with the databases that correspond to the parameters `database` and `database_[n]`.	Local hostname
`dmuser` `dmuser_[n]`	Specifies the deployment queue service user account. This is a registered Dimensions user that is used to read and initiate deployment job processing in databases that correspond to the parameters: ■ `database` ■ `database_[n]` ■ `host` ■ `host_[n]` You must register the passwords corresponding to the Dimensions users in `DM_ROOT/dfs/registry.dat` (use the dmpasswd utility).	dmsys
`threads` `threads_[n]`	Specifies the number of threads that will be used to execute deployment jobs in the databases that correspond to the parameters: ■ `database` ■ `database_[n]` ■ `host` ■ `host_[n]`	20

Examples

- This example is a simple deployment server configuration for the default cm_typical installation on the host `dimsrv`:

```
database=cm_typical@dim10
host=dimsrv:671
dmuser=dmsys
```

- This example is deployment server configuration that monitors two base databases on the same host:

```
database_1=devtest@dim10
host_1=dimsrv:671
dmuser_1=dmsys

database_2=production@dim10
host_2=dimsrv:671
dmuser_2=dmsys
```

- This advanced example is a deployment server configuration that monitors three base databases on different hosts with logging turned on:

```
log_dir=c:/temp/
idle_timeout=10

database_1=devtest@dim10
host_1=dimsrv:671
dmuser_1=dmsys
threads_1=5

database_2=production@dim10
host_2=dimprodsrv:672
dmuser_2=dmadmin
threads_2=20

database_3=devtest2@dim10
host_3=dimsrv:671
dmuser_3=dmsys
threads_3=10
```

Chapter 10

Dimensions Deployment Privileges

Deployment Privileges

This appendix lists all of the deployment related privileges. For a detailed description of the rules that apply to each privilege see the *Privilege Rules* section in the *Dimensions CM Privileges* appendix in the *Process Configuration Guide*.

Item Privileges

Privilege Name	Privilege	Description
Deploy to Areas	ITEM_DEPLOY	This privilege allows you to deploy an item to a deployment area.
Rollback from Areas	ITEM_ROLLBACK	This privilege allows you to roll back an item from a deployment area.
Promote to Any Stage	ITEM_ PROMOTE _ANYSTAGE	This privilege allows you to promote an item to any promotion stage in the lifecycle.
Promote to Next Stage	ITEM_ PROMOTE _NEXTSTAGE	This privilege allows you to promote an item to the next promotion stage in the lifecycle.
Demote to Any Stage	ITEM_DEMOTE_ANYSTAGE	This privilege allows you to demote an item to any demotion stage in the lifecycle.
Demote to Next Stage	ITEM_ DEMOTE _NEXTSTAGE	This privilege allows you to demote an item to the next demotion stage in the lifecycle.

Request Privileges

Privilege Name	Privilege	Description
Deploy to Areas	REQUEST_DEPLOY	This privilege allows you to deploy a request to a deployment area.
Promote to Any Stage	REQUEST_PROMOTE_ANYSTAGE	This privilege allows you to promote a request to any promotion stage in the lifecycle.
Promote to Next Stage	REQUEST_PROMOTE_NEXTSTAGE	This privilege allows you to promote a request to the next promotion stage in the lifecycle.
Demote to Any Stage	REQUEST_ DEMOTE _ANYSTAGE	This privilege allows you to demote a request to any demotion stage in the lifecycle.
Demote to Next Stage	REQUEST_ DEMOTE _NEXTSTAGE	This privilege allows you to demote a request to the next demotion stage in the lifecycle.

Baseline Privileges

Privilege Name	Privilege	Description
Deploy to Areas	BASELINE_DEPLOY	This privilege allows you to deploy a baseline to a deployment area.
Promote to Any Stage	BASELINE_ PROMOTE _ANYSTAGE	This privilege allows you to promote a baseline to any promotion stage in the lifecycle.
Promote to Next Stage	BASELINE_ PROMOTE _NEXTSTAGE	This privilege allows you to promote a baseline to the next promotion stage in the lifecycle.
Demote to Any Stage	BASELINE_ DEMOTE _ANYSTAGE	This privilege allows you to demote a baseline to any demotion stage in the lifecycle.
Demote to Next Stage	BASELINE_ DEMOTE _NEXTSTAGE	This privilege allows you to demote a baseline to the demotion stage in the lifecycle.

Chapter 11

Deployment and Refactoring

Overview

The purpose of this appendix is to explain how to deploy refactoring changes using requests. It also contains a number of short scenarios to illustrate what happens for the basic types of refactoring changes. These scenarios complement the detailed refactoring scenario, "Scenario 4: Deploying Refactoring Changes" on page 144.

What is Refactoring?

Refactoring occurs when there are modifications within the project/stream structure such that the location of items or folders are changed. Examples would be exporting an item to or from a project, or moving a project/stream folder from one parent folder to another. Operations that result in refactoring are:

- File renames
- File moves
- Folder renames
- Folder moves
- File removals
- Folder removals.

When you make these changes in Dimensions CM clients (web client, desktop client, or dmcli client) these changes will only happen automatically in:

- The Dimensions CM project/stream in which you perform the changes
- Any areas associated with the initial deployment stage for the project that are defined as *Deploy by Default*.

In order to have these changes reflected in the deployment areas for other stages in the GSL (global stage lifecycle) you will need to take some additional action to deploy these changes.

Request Driven Refactoring

Using requests to track refactoring changes is the recommended method of ensuring that these changes are reflected in the appropriate deployment areas. Baselines cannot be used to deploy refactoring changes.

When you perform actions that will result in refactoring, such as exporting a file to a project, or renaming a project/stream folder, those changes are recorded or tracked in Dimensions CM against a request ID, provided a request was specified when those actions were performed. When you subsequently promote one of these requests to another stage in the GSL and deploy it, the refactoring changes that were tracked against that request will also be applied to the deployment areas associated with that stage to which it has been promoted.

For example, a developer needs to move a file `foo.c` from the folder `src` to the folder `utils`. The project uses the GSL consisting of the stages:

DEV -> SIT -> QA -> LIVE

1 The developer uses the desktop client Move dialog box to move the item in the project, and enters a request, R1, in the **Track changes with request(s)** field.

   The project and the areas associated with the DEV stage now contain the item file:

   ```
   utils/foo.c
   ```

   Whereas the deployment areas for stages SIT, QA, and LIVE still contain the file:

   ```
   src/foo.c
   ```

2 He then wants to deploy this change to SIT. He therefore promotes and deploys the request R1 to the SIT stage.

   The deployment areas for SIT now contain the file `utils/foo.c`. The deployment areas for QA and LIVE, however, still contain the file `src/foo.c`.

3 When the change is ready to be deployed to QA and LIVE, the request R1 is subsequently promoted and deployed to those stages.

Request driven refactoring has the advantage of supporting all the types of refactoring: removing, moving, or renaming of folders or files. It also does not present a problem when demoting back down the GSL.

NOTE When using request driven deployment, refactoring changes must be deployed up in the order in which the refactoring changes were carried out. For example, if the file `foo.c` is added to the project, and then it is renamed to `bar.c`, the file `foo.c` needs to have been promoted to a particular stage before the rename can be promoted to that stage.

Deployment of Refactoring Changes

How do you Deploy Refactoring Changes?

In order for request deployment of refactoring changes to be performed, it is necessary to make sure the refactoring changes have been tracked by a request. When you perform actions that will result in refactoring, such as exporting a file to a project, or moving a project folder, those changes will be recorded against the request ID that you supplied when you performed that action. For refactoring operations to occur in areas associated with stages other than the initial stage, you must promote that request to the required stage and then the refactoring changes will be reflected in the areas associated with that stage when the request is deployed.

For projects, it is advisable to set the project option:

Request required to refactor

This means that a user is required to provide one or more request IDs when they perform an action that results in refactoring, such as exporting an item or renaming a project folder, otherwise the action will not be completed.

For streams, it is advisable to set the option:

Valid request must be specified when delivering changes

In the case of streams, refactoring changes take place in the work area and are then delivered to the stream. Using the request that was specified when the changes were

delivered means that the request can be used to deploy any refactoring changes that occurred.

Example Deployment Scenarios

This section details a number of example deployment scenarios for refactoring changes and explains in detail what occurs for each example. The examples illustrate the following scenarios:

- File rename
- File move
- Folder rename
- Folder move
- File remove
- Folder remove

Deployment Stages Used in the Examples

These examples assume a global stage lifecycle with four stages, "DEV" (Development) "SIT" (System Integration Test) "QA" (Quality Assurance) and "LIVE", each of which have a single attached deployment area, but the same principles would also apply with multiple deployment areas. It is assumed that these deployment areas are attached to the project as *Deploy by Default,* so that the files are automatically deployed to the area whern they are promoted to the corresponding stage. Filenames are shown in UNIX format (using forward slashes) and the item revision numbers are denoted by a semi-colon followed by the revision number.

The examples assume that the areas involved are associated as *Deploy by Default* for the project concerned, so that the deployments occur automatically when the changes are promoted. They also assume projects are being used.

Deployment of Regressions

By default, a higher revision of an item file will replace a lower revision if it is deployed to an area, but this behavior is optional. For details, see *Deploying Regressions* in the *Deployment Guide*.

Scenario for Renaming a File

When you rename a file in a Dimensions project with attached deployment areas, Dimensions CM will also rename the file on disk in any deployment areas associated with the first stage as default areas. The file will continue to use its original name in areas associated with other stages.

When you promote this change to another stage, the rename will take effect in the deployment areas for that stage. At that time the old file will be removed from those areas and the file will be placed in the areas using the new name.

Example: Content and name change

This example illustrates what will happen if you rename a file in a project and also change its contents. Imagine a Dimensions project contains a file called "src/main.java" and there are various revisions of that file promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1

The file names are currently all the same in each area and earlier revisions have progressed further along the stage lifecycle than newer revisions.

A user then refactors the java code and renames "src/main.java" to "src/mainevent.java" (changing its content and name). Only the project and the DEV area are affected so we now have:

Project	DEV	SIT	QA	LIVE
src/mainevent.java;5	src/mainevent.java;5	src/main.java;3	src/main.java;2	src/main.java;1

If these changes are promoted via a request to “SIT”, the project and area contents would now be:

Project	DEV	SIT	QA	LIVE
src/mainevent.java;5	src/mainevent.java;5	src/mainevent.java;5	src/main.java;2	src/main.java;1

So as you can see the new revision was placed in "SIT" and the rename took effect.

Example: Name change only

Note that, in the example above when the change was deployed, revision 3 was at SIT and was replaced by the new content (revision 5) in addition to having its name changed. It is not however necessary to change content for a file rename to be deployed. The following example shows a rename without content change being deployed.

The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
docs/login.html;4	docs/login.html;4	docs/login.html;4	docs/login.html;2	docs/login.html;1

The file names are currently all the same in each area, and earlier revisions have progressed further along the stage lifecycle than newer revisions.

Imagine the user then uses Desktop Client to rename `docs/login.html` to `docs/welcome.html`. Only the project and the initial area are affected so we now have:

Project	DEV	SIT	QA	LIVE
docs/welcome.html;4	docs/welcome.html;4	docs/login.html;4	docs/login.html;2	docs/login.html;1

Next the change is promoted to SIT, the project and area contents would now be:

Project	DEV	SIT	QA	LIVE
docs/welcome.html;4	docs/welcome.html;4	docs/welcome.html;4	docs/login.html;2	docs/login.html;1

So as you can see the rename correctly took effect in the SIT area despite the revision already being at the SIT stage.

Scenarios for Moving a File

When you move a file in a Dimensions project with attached deployment areas, Dimensions CM will also move the file on disk in any default deployment areas associated with the first stage. The file will continue to reside in its original location in areas associated with other stages.

When you promote this change to another stage, the move will take effect in the deployment areas for that stage. At that time the file will be removed from the original folder and will be placed in the new folder in areas for the selected stage.

Example: File move

Imagine that a Dimensions CM project contains two folders. One folder is called `src` and contains two files (`src/main.java` and `src/trace.java`). The other folder is called `utils` and only contains the file `logging.java`, and there are various revisions of these files promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1
utils/logging.java;7	utils/logging.java;7	utils/logging.java;6	utils/logging.java;5	utils/logging.java;4

The file names and folder locations are currently all the same in each area and earlier revisions have generally progressed further along the stage lifecycle than newer revisions.

Imagine that the user then moves `src/trace.java` to the `utils` folder. Only the project and the initial area are affected so we now have:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
utils/trace.java;2	utils/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1
utils/logging.java;7	utils/logging.java;7	utils/logging.java;6	utils/logging.java;5	utils/logging.java;4

Next, a request is used to track the move, and that request is promoted to SIT. The project and area contents would now be:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1

Project	DEV	SIT	QA	LIVE
utils/trace.java;2	utils/trace.java;2	utils/trace.java;2	src/trace.java;1	src/trace.java;1
utils/logging.java;7	utils/logging.java;7	utils/logging.java;6	utils/logging.java;5	utils/logging.java;4

Note that if all of the files in the `src` folder were moved into the `utils` folder and promoted to SIT then the empty folder `src` would remain on disk in your deployment areas. There is no mechanism provided for the removal of empty folders from deployment areas.

Scenarios for Renaming a Folder

When you rename a folder in a Dimensions project with attached deployment areas, Dimensions CM will also rename the folder on disk in any default deployment areas associated with the first stage. The folder will continue to use its original name in areas associated with other stages.

When you use a request to track the rename, and you promote that request to another stage, the rename operation will take effect in the deployment areas for that stage. The files being deployed will then appear in the new folder.

Example: Folder rename

Imagine that a Dimensions CM project contains a single folder called `src` containing two files (`src/main.java` and `src/trace.java`). There are various revisions of these files promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

The file names and folder locations are currently all the same in each area and earlier revisions have generally progressed further along the stage lifecycle than newer revisions.

Imagine that a user then renames `src` to `source`. Only the project and the initial area are affected so we now have:

Project	DEV	SIT	QA	LIVE
source/main.java;4	source/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
source/trace.java;2	source/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

Next, you promote a request referencing the change to the folder name to SIT. The project and area contents would now be:

Project	DEV	SIT	QA	LIVE
source/main.java;4	source/main.java;4	source/main.java;4	src/main.java;2	src/main.java;1
source/trace.java;2	source/trace.java;2	source/trace.java;2	src/trace.java;1	src/trace.java;1

The `src` folder in the SIT area now contains no Dimensions CM controlled files.

Scenarios for Moving a Folder

When you move a folder in a Dimensions project with attached deployment areas, Dimensions CM will also move the folder on disk in any default deployment areas associated with the first stage. The folder will continue to use its original location in areas associated with other stages.

When you use a request to track the move, and you promote that request to another stage, the move operation will take effect in the deployment areas for that stage. The files being deployed will then appear in the new folder location.

Example: Folder move

Imagine that a Dimensions project contains two folders. One folder is called `src` and contains two files (`src/main.java` and `src/trace.java`). The other folder is called `utils` and only contains the file `log.java`, and there are various revisions of these files promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1
utils/log.java;7	utils/log.java;7	utils/log.java;6	utils/log.java;5	utils/log.java;4

The file names and folder locations are currently all the same in each area, and earlier revisions have generally progressed further along the stage lifecycle than newer revisions.

A user then moves the `utils` folder so that it becomes a child of the `src` folder. Only the project and the initial area are affected, so we now have:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;3	src/main.java;2	src/main.java;1
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1
src/utils/log.java;7	src/utils/log.java;7	utils/log.java;6	utils/log.java;5	utils/log.java;4

Next, a request tracking the change, is promoted to SIT, The project and area contents will now be:

Project	DEV	SIT	QA	LIVE
src/main.java;4	src/main.java;4	src/main.java;4	src/main.java;2	src/main.java;1
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1
src/utils/log.java;7	src/utils/log.java;7	src/utils/log.java;7	utils/log.java;5	utils/log.java;4

The `utils` folder in the root of the SIT area now contains no Dimensions CM controlled files.

Scenarios for Removing a File

When a file is removed from a Dimensions CM project with attached deployment areas, Dimensions CM will remove the file from any default deployment areas associated with the first stage. The file will continue to exist in areas associated with other stages.

When a file is removed and a request is specified, simply promoting that request to a stage will result in the file being removed from the deployment areas for that stage.

Example: Deploying removal of an item revision

A Dimensions project contains a folder called `src` containing a file called `src/trace.java`. Various revisions of the file have been promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

Next, a developer decides to remove `trace.java;2`, as perhaps it introduced a bug and they wish to quickly back it out. They do this using the Remove Item from Project dialog box in the desktop client, specifying a request R1 in the *Track changes with requests(s)* field.

At this point, the contents of the project and associated areas will look like this:

Project	DEV	SIT	QA	LIVE
src/trace.java;1	src/trace.java;1	src/trace.java;2	src/trace.java;1	src/trace.java;1

Request R1 is then promoted to the SIT stage. Revision 2 of `src/trace.java` is now removed from SIT:

Project	DEV	SIT	QA	LIVE
src/trace.java;1	src/trace.java;1	src/trace.java;1	src/trace.java;1	src/trace.java;1

Example: Deploying removal of an item revision using baselines

As above, a Dimensions project contains a folder called `src` containing a file called `src/trace.java`. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

Next, a developer decides to remove `trace.java;2`. They do this against a request and relate `trace.java;2` as *Affected* to mark it as a removal. They create a revised baseline and specify this request in the "Remove" request list. This will create a baseline that no longer contains "trace.java;2". At this point, the contents of the project and associated areas will look like this:

Project	DEV	SIT	QA	LIVE
src/trace.java;1	src/trace.java;1	src/trace.java;2	src/trace.java;1	src/trace.java;1

When this baseline is prompted to SIT, revisions of the items related as "Affected" to the requests specified in the "Remove" list for the revised baseline will be removed from the SIT deployment areas.

The fact that the item revision no longer exists in the project does not affect this. Revision 2 of `src/trace.java` is now removed from SIT:

Project	DEV	SIT	QA	LIVE
src/trace.java;1	src/trace.java;1	src/trace.java;1	src/trace.java;1	src/trace.java;1

Example: Deploying removal of all item revisions using requests

In this example we will see how you can remove all revisions of an item from a deployment area. Imagine a Dimensions project containing a single folder called `src` containing the file `src/trace.java`. Various revisions of the file have been promoted to different stages. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

Next a developer decides to remove all revisions of `trace.java` as perhaps its functionality was moved into another file. They specify a request when removing both the revisions of `trace.java` using the Remove Item from Project dialog box in the desktop client.

At this point in time the project and area contents will look as follows:

Project	DEV	SIT	QA	LIVE
		src/trace.java;2	src/trace.java;1	src/trace.java;1

When the request is promoted to SIT, the project and area contents will now look as follows:

Project	DEV	SIT	QA	LIVE
			src/trace.java;1	src/trace.java;1

All revisions of `src/trace.java` have now been removed from "SIT".

Example: Deploying removal of all item revisions using baselines

Now we will see how you can remove all revisions of an item from a deployment area when the deployment method for the project is *baseline*. Imagine the same example of a Dimensions project containing a single folder called `src` containing the file `src/trace.java`. Various revisions of the file have been deployed to different stages using baselines. The following table shows what is in the main project and each area:

Project	DEV	SIT	QA	LIVE
src/trace.java;2	src/trace.java;2	src/trace.java;2	src/trace.java;1	src/trace.java;1

A developer decides to completely remove all revisions of `trace.java`. As described above, if you are using baseline deployment, removals are only deployed if they are related as "Affected" to a request used in the "Remove" list for a revise baseline

operation, and that baseline is promoted. So the developer relates both revision 1 and 2 of `src/trace.java` as "Affected" against a request, and creates a revised baseline specifying that request in the "Remove" request list. This will create a baseline that no longer contains `trace.java`. They also remove both revisions of the item from the project.

At this point in time the project and area contents will look as follows:

Project	DEV	SIT	QA	LIVE
		src/trace.java;2	src/trace.java;1	src/trace.java;1

When the baseline is promoted to SIT, the project and area contents will now look as follows:

Project	DEV	SIT	QA	LIVE
			src/trace.java;1	src/trace.java;1

All revisions of `src/trace.java` have now been removed from "SIT".

Folder Removal

If the developer specifies a request when deleting an empty folder, then when that request is promoted to a stage, the empty folder will be deleted from the corresponding areas.

Rollback of Deployed Objects to an Earlier Stage

The Global Stage Lifecycle (GSL) allows for transitions back to earlier stages in the lifecycle. A version of an area can be rolled back provided there are no subsequent versions of the area that depend on any changed made by the version being rolled back. Refactoring changes must be demoted in the reverse order to which they were deployed up.

Scenarios for Demoting

If you are demoting a request deployment, then file and folder removes will revert to their old values when the request is demoted.

Example: Demoting a file rename

This example illustrates what will happen if you demote a previously promoted file rename.

The following table shows the filename and revisions initially in the main project and each area. The filename is the same, and earlier revisions have progressed farther along the stage lifecycle with later revisions:

Project	DEV	SIT	QA	LIVE
Docs/login.html;4	Docs/login.html;4	Docs/login.html;4	Docs/login.html;2	Docs/login.html;1

The user uses the desktop client to rename the file in the project to `welcome.html` against the default request R1, and the filename is changed in the project and the initial stage:

Project	DEV	SIT	QA	LIVE
Docs/welcome.html;4	Docs/welcome.html;4	Docs/login.html;4	Docs/login.html;2	Docs/login.html;1

This change is then deployed to SIT by promoting request R1, and the project and area contents become:

Project	DEV	SIT	QA	LIVE
Docs/welcome.html;4	Docs/welcome.html;4	Docs/welcome.html;4	Docs/login.html;2	Docs/login.html;1

So the file is now renamed in the SIT area.

Now, at this point testing occurs at the SIT stage, and the team realizes that the file rename needs to be backed out from the SIT stage. The team leader demotes request R1 back to DEV, and the project and area contents become again:

Project	DEV	SIT	QA	LIVE
Docs/welcome.html;4	Docs/welcome.html;4	Docs/login.html;4	Docs/login.html;2	Docs/login.html;1

So the file rename was reverted in the SIT area.

Index

E

F

G

H

I

J

L

M

N

P

Q

R

S

T

V