

---

# IBM Rational ClearCase and IBM Rational ClearQuest Evaluation Guide, Part 1: Unified Change Management

An introduction to IBM's software change and configuration management solution

Skill Level: Intermediate

[Paul Boustany \(pboustan@us.ibm.com\)](mailto:pboustan@us.ibm.com)  
Software engineer  
IBM

31 Oct 2006

This tutorial is intended to serve as a guide for an [online demonstration](#) of IBM® Rational® Unified Change Management (UCM). [Part 2](#) of this series is intended to help you through a [demo of IBM Rational ClearQuest®](#), IBM Rational's change management solution. [Part 3](#), guides you through the [online demonstration](#) for IBM Rational ClearCase®.

## Section 1. Before you start

In this tutorial, you will learn some of the basic tasks for using Rational ClearCase and Rational ClearQuest for team software development, and the organizational roles associated with those actions. You will learn how these tools are used by project managers to assess and assign changes, by developers to implement new changes and deliver them to the team, and by build and quality managers to test, promote and release new builds.

## About this series

This series is for users interested in trying some of the change and configuration management capabilities of Rational ClearCase and Rational ClearQuest without installing or configuring the tools. This evaluation provides both tools in a ready to use state from inside a single Eclipse-based shell. By the end of the evaluation, you should have a feel for how Rational ClearCase and Rational ClearQuest work together to provide a complete change and configuration management solution.

## About this tutorial

This tutorial provides a basic introduction to Unified Change Management using Rational® ClearCase® and ClearQuest® to manage the software development lifecycle. [Part 2](#), presents an overview of Rational ClearQuest and how it is used alone to manage defects, enhancements and other changes. [Part 3](#) describes how to perform the basic tasks within Rational ClearCase.

## Objectives

In this tutorial you will be fulfilling the role of several stakeholders in the software development process including: project lead, developer, integrator and release engineer. By acting in these roles, you will work a defect through the development lifecycle; from submission to completion. Along the way you will observe how Rational ClearCase and Rational ClearQuest work together to control and manage the process of change in software development.

## Prerequisites

Basic knowledge of Rational ClearQuest and/or Rational ClearCase or configuration management concepts is helpful, but not required. Once you start the trial, you have three hours to explore the application. We recommend that if you are new to Rational ClearCase and/or Rational ClearQuest, you use our evaluation guide to explore some core features.

## System requirements

The online trial program uses the Citrix® Access Gateway™ platform to provide you a connection from your workstation to a remote server running the IBM product you are evaluating. You will need to download Citrix client software before using the online trial program. You can download the Citrix MetaFrame® Presentation Server™ client at no charge from the [Citrix site](#); many versions, including versions for Windows® and Linux® platforms, are available. After you install the client, you will be asked to restart your browser. If you do not have the Citrix client installed when you attempt to access the online trial, you will be prompted to install the client.

---

## Section 2. Unified Change Management: Project lead role

You will start this evaluation in the role of the Project Lead. The project, organized by the Project Lead, is broken down into multiple components. Within those components are different required tasks. The basic tasks for this role include:

- Logging on to the ClearQuest client
- Running a query and viewing a distribution chart
- Modifying a record and assigning a new owner for a defect

### Log on and run a query

1. In the ClearQuest perspective, click the **ClearQuest** menu and then click **Login > Login**.

#### Figure 1. Logging in to ClearQuest

#### Figure 2. The ClearQuest Navigator

On the left, there is a menu of different queries available. You can use one of these or create your own custom queries to fit your needs.

2. In the navigation pane, expand **Public Queries > Classics PointOfSale Project > Submitted High Severity Defects**.

Performing bug triage has never been easier. By running saved queries, the Project Lead can quickly identify which change requests or defects need immediate attention. And by looking at distribution charts, you can identify the workloads of your engineers and assign the defects to those that have the cycles to resolve them.

### Viewing a chart of defects by owner

From the left-hand menu, you select **Submitted High Severity Defects**. These are the defects that have been submitted but not yet opened by Engineering. In the **ClearQuest Query Results** pane, you see that the defect report with the headline: **Fix spelling error in Readme.html** has yet to be assigned, that is, It is still in the **Submitted** state.

### Figure 3. Chart of Defects by Priority by Owner

Next, you run the query **Distribution Charts > Defects by Priority > Owner** which brings up a chart of the defects assigned to each engineer. You can use this information to decide to whom this defect can be assigned.

3. Click on **Public Queries > ClassicsPointOfSaleProject > Distribution Charts > Defects by Priority > Owner**

On the chart shown in Figure 3, you see that team member *Alex* only has a few activities that are in the **Opened** or **Assigned** state, so you can assign this defect to Alex.

## Modifying a record: Assign a new owner

Now that you have identified a developer to be the owner of the defect, you can modify the record and set the appropriate fields on the change request form. All the fields on a Rational ClearQuest form are completely customizable. You execute the **Assign** action and enter *Alex* as the **Owner**. You also set the **Priority** field and enter a note saying to further explain the change request. Once you've recorded all this information, you click **Apply** to update the record.

1. In the Query results pane of ClearQuest Navigator (Figure 2), right-click on record **CLSIC00000058**, and click **Change State > Assign**.
2. In the **Modify...(record)** dialog box, select the **Main** tab.
3. Under **Priority**, select **1 - Resolve immediately**.
4. Under **Owner**, select **alex**.

### Figure 4. Modify (record) dialog box - Main tab

5. Click the **Notes** tab.
6. Under **New Note**, type some meaningful text for the new owner.
7. Click in the **Notes Log** area to see you note added to this field.

Note: You can click the icon next to the **New Note** and **Notes Log** fields to bring up a text editor (Figure 5).

### Figure 5. Modify (record) dialog box - Notes tab

ClearQuest hooks can fire when an action is performed. One very important hook is the email notification hook. In this example, once the *Modify* action is complete, the

notification hook could execute any *Email Rules* associated with the change request. For example, you may want the email rules set to watch for changes made to the *Owner* field and when the value changes, automatically generate and send email to the new owner informing them of the change.

8. Click the **Audit Trail** tab.
9. Verify that all the the audit information is captured in this record.
10. Click **OK** to save your changes.

To meet regulatory compliance requirements, ClearQuest keeps a record of every action performed on a defect. The audit trail tab automatically keeps this information for the entire life of the defect.

### Verify your changes and log out

You can now re-run the Defects by Priority > Owner chart and confirm that Alex now has ownership of the Submitted defect that you just modified.

1. Click on **Public Queries > ClassicsPointOfSaleProject > Distribution Charts > Defects by Priority > Owner**
2. Once you have confirmed your changes, click on the **ClearQuest** menu, select **Database**, and then click **Disconnect**.
3. Logout of ClearQuest as user *pat* but remain in the ClearQuest perspective for the next section (where you will log back in as *alex*).
4. Click **OK** to confirm your selections.

In summary, in this section we have seen how the Project Lead can:

- Assess the project status and progress through standard and customized queries and charts.
- Effectively allocate change requests to engineering resources.
- Use customizable solutions or hooks to assign changes requests and automatically communicate project status to the team

---

## Section 3. Unified Change Management: Developer role

As a developer, identifying and managing the work you are expected to do and informing others of the work you have done can be one of the biggest challenges in team-based development. With strong organization and communication channels, the development team can stay in synch and react quickly to new information.

Now you can use ClearQuest directly from within the Eclipse environment to review the tasks you need to do. ClearQuest makes creating reports easy with simple to use wizards. You can easily make personal or public queries on any record type in the database.

In this section you will learn how to:

- Create a To Do list from a query
- Join a Unified Change Management Project
- Deliver an activity to an integration stream

Let's take a look at the querying functions available in Rational ClearQuest. We will start by creating a new query for the developer *Alex* that will run each time he logs into ClearQuest. This query will show him his 'To Do' list.

## Log on and run a query

1. In the ClearQuest perspective, click the **ClearQuest** menu and then click **Login > Login**.
2. Log in as user/password *alex/alex*.
3. Right click on **Personal Queries** and select **New Query**.

### Figure 6. Starting a new personal query

4. From the Query Wizard, select the record type **Defect** and give the query a name like *Alex\_defects*, and click **Next**.

### Figure 7. Selecting the defect record type

5. Now pick the fields you want to filter your query. From the left pane, double click **Headline**, **ID**, **Owner** and **Priority**, and click **Next**.

### Figure 8. Selecting the fields for query filters

6. Now select the value to filter on. Expand the **And** menu, highlight **Owner** and click the **Values** button.

### Figure 9. Selecting the query values

7. In the **Select Values** pop-up window, select a value for the **Owner** field: Put a check mark next to **Alex** then click **OK**.

### Figure 10. The Select Values dialog box

8. In the **Define Query Filters** step of the Query Wizard, click **Next**.
9. Now you will define which fields and the order in which to display them. In the **Define Display Fields** step of the wizard, in the following order, select:
  - **Priority**
  - **Headline**
  - **ID**
  - **Owner**
11. Select the **Run Query** check box, and then click **Finish**.

### Figure 11. Defining the Display Fields

Once your query is defined, you can have it run immediately after finishing the Wizard. Based on the query we just defined, the developer *Alex* can now quickly discover any new work that he may have. It looks like the project lead *Pat*, just assigned Alex some new work. Let's take a look at what is needed to fix this defect.

11. Double click on defect **CLSLIC00000058, Fix spelling error in Readme.html**, which Pat just assigned to Alex.
12. Review the details of this defect, including the note that *Pat* left on the **Notes** tab, and close the defect form.

Important information can be shared with your team members easily with ClearQuest forms. All your notes get stored in a secure database, so you can rebuild a complete history of every defect that's been worked on.

13. Disconnect from ClearQuest as user *Alex*.

## Join a UCM project

IBM Rational ClearCase and Rational ClearQuest change management products work together to allow you to define and manage changes to software assets as

activities. Through the Unified Change Management (UCM) capability, file versions in Rational ClearCase software are associated with activities in Rational ClearQuest. Developers can then perform operations directly on the activities rather than on the collections of files associated with them. This activity-based approach to change and configuration management helps developers to work on the right versions of the right files. It allows them to create collections of file versions, or "change sets," for each individual change. This enables developers to manage their work at the task level, instead of managing individual files. It helps build engineers ascertain that the right files are incorporated into the build. Testers can easily confirm that the right functionality and builds are tested. Quality assurance (QA) engineers can quickly see and validate what has changed between builds. And project managers can more easily track and assess project status.

1. Click on the ClearCase perspective. Click the **Join a UCM Project** button on the Toolbar.

#### Figure 12. Joining the UCM project

2. In the **Join UCM Project** wizard, under **ClearCase Web server URL**, enter: `http://localhost/ccrc`
3. Enter **alex/alex** in the **User Name/Password** fields, and click **Next**.

#### Figure 13. Selecting a ClearCase Web server

4. Under **Select a UCM project to join**, expand the **CLSICS\_pvob** menu, select the **UCM Project CLSICS\_CD** and click **Next**.

#### Figure 14. Selecting a project

5. In the **Specify UCM Streams** step of the wizard, select the checkbox to **Create a new development stream**. Under **Stream name**, enter **alex\_task\_stream**, and click **Next**.

#### Figure 15. Creating a new project stream

6. In the **Create a ClearCase UCM development view** step, under **Copy area path name**, enter the path to the view copy area : **C:\Documents and Settings\Administrator\alex\_task\_stream**, and click **Next**.

#### Figure 16. Creating a development view

7. In the **Create a ClearCase UCM integration view** step, select the checkbox to **Create a ClearCase integration view**. Under **Copy area path name**, name the view copy area: **C:\Documents and Settings\Administrator\alex\_CLSICS\_CD\_intg**, and click **Finish**.

### Figure 17. Creating an integration view

8. When prompted to use the **Clearcase View Configuration tool** to load the views, click **Yes**.

### Figure 18. Loading the views

9. In the **ClearCase View Configuration** dialog box, Click the checkbox to **Show all VOBs (includes private VOBs)**, then select the root of the **CLSICS\_comp1** VOB and click **OK**.

### Figure 19. Adding the load rules for the view

After choosing your view's configuration, you must select which assets you want to work on. Once the selection is made, your local workspace becomes populated with those files and directories.

### Figure 20. Loading the files to the view

Now we want to find the activity that we are going to be working on.

10. In ClearCase Explorer, click on **My Activities** under the newly created 'task' view.

### Figure 21. Activities associated with the view

#### Working on My Activities

Because our UCM project is linked to ClearQuest, we need to login to the database to get a list of available defects.

1. Log in to the ClearQuest database using username/password *alex/alex*.

### Figure 22. Logging in to ClearQuest

2. Select the activity **Fix spelling error in Readme.html**, right-click, and click **Work on Activity**.

### Figure 23. Selecting an activity to work on

To verify the defect, navigate to Readme.html (in the ClearCase Details pane) by double-clicking on **CLSICS\_comp > MyHomePage > WebContent**. Right-click on the file **Readme.html** and click **Open**. You should see the file displayed in a browser window, and verify the two spelling errors in the headline. Close the browser window.

3. Open a **Resource** perspective by clicking on the **Open a perspective** button in the upper right corner of the ClearCase perspective, and clicking

## Resource.

### Figure 24. Opening a Resource perspective

4. In the Resource perspective, click the File menu and then click Import

### Figure 25. Importing the project

5. Under **Select an import source**, select **Existing project into Workspace** and click **Next**.

### Figure 26. Selecting an import source

6. Click the **Browse** button and navigate to the **MyHomePage** directory, (path: C:\Documents and Settings\Administrator\alex\_task\_stream\CLSICS\_comp1\MyHomePage), click **OK**, and click **Finish**.

### Figure 27. Importing the project Editing Readme.html

1. Navigate to the file `Readme.html` (found in the `WebContent` directory under `MyHomePage`).

### Figure 28. Importing the project

2. Find the line starting '<h2 ' From here you can see the spelling errors that need to be corrected.
3. Try to delete the extra 'd' in the word 'Readme.'

The application is aware that the file you are editing is under source control. Any attempts to modify it, will prompt you to check the file out.

4. In the Checkout Resources dialog box, click Apply to check the file out. (Note that the activity you chose to work on in the previous step, is selected by default.)

### Figure 29. Checking out the file

5. Correct the spelling errors by removing the extra 'd' in 'Readme' and the 'i' from 'Home.'
6. Save the changes by clicking the **Save** button.

Typically after a developer makes changes to any files, he will want to test those

changes. For the sake of this evaluation, we won't need to test our changes, but let's at least verify that we did correct the defect.

7. Navigate back to the ClearCase perspective.
8. Right-click on the file again and select **Open**. Verify that the defect is fixed.

The changes look good! Let's make our changes complete by checking the file back in to the repository.

9. Close the web browser window displaying the file.
10. Right click on `Readme.html` and select **Checkin**. Accept all defaults and click **Apply**. (You can use the same comment to check in the file as you did to check it out.)

### Figure 30. Checking the file back in

Now the file is checked in to the VOB.

## Delivering the activity to the integration stream

While we've made the necessary changes in our own work area, the changes haven't yet been propagated to the project. Moving changes made for an activity (or activities) from a development work area to a project integration work area is called a *Deliver* operation.

Delivering changes to the project only when appropriate enables developers to work independently in parallel with each other while maintaining project stability.

Now let's deliver our completed activity to the project.

1. Click on Alex's development view. Then click on the **Deliver Stream** icon in the ClearCase toolbar at the top of the screen.

### Figure 31. Deliver button

2. In the Deliver from Stream window, be sure to check the Merge elements graphically checkbox.

### Figure 32. Delivering from a stream

3. Click the **Detail** button to confirm the activities being delivered.

4. Click **Change** to change the view to select the proper view target. Select **alex\_CLSICS\_CD\_intg** and click **OK**.
5. Click **OK** to begin delivering the change.

### Figure 33. Selecting the view

Rational ClearCase uses the information associated with your development stream to determine which activities have been worked on and are ready to be delivered. Based on that information, it knows which files need to be merged to your project's Integration stream.

As long as there are no conflicts between your changes and changes made by other developers on your project, the delivery will be completed without any further intervention. In this example, there is a conflict.

In cases where there are conflicts that require you to intervene, Rational ClearCase will walk you through the process of resolving the conflicts.

6. In the **Deliver** dialog box, click **Yes** to start the merge tool

First, you select the option to use the graphical merge tool to resolve the conflicts. You show both changes to see what the differences are. You see that there is a conflict between the changes that you made and the version that was in your project's integration stream. If you examine the difference however, you can see that the change that you made in your private development workspace are in fact the correct changes, so we will want to accept those changes. To complete the merge, you save the file and exit the merge tool.

### Figure 34. Starting the merge

7. Click **OK** in the Diff Merge dialog box to begin the merge.

### Figure 35. Resolving the differences

8. Contributor 2 has the change that was made. Click the button **2** on the toolbar to accept that change.

### Figure 36. Confirming the changes

9. Confirm that the resulting change is correct, and click the **Save** button.
10. If prompted, click **Yes** to replace the output file.
11. Close the merge tool.

12. Click **OK** in the confirming **Deliver** dialog box.

### Figure 37. Confirming the merge

13. The deliver results are now displayed.

### Figure 38. Results of deliver operation

14. Navigate to the integration view that was used as the deliver target and open the `Readme.html` file to verify that the change made it to the integration view. Close the web browser.
15. Click the **Complete Deliver** button in the delivery summary pane to finish the deliver operation.

### Figure 39. Completing the deliver

16. Click the **Resolution** tab on the ClearQuest web form.
17. Pick the resolution **Fixed** from the drop-down and click **Save**.
18. Close the ClearQuest Web form.

Before completing any deliver operations, teams should ensure that the changes that were delivered are good, and don't impact other work in the integration area. At this point, most teams would want to run a build or test to ensure that everything will still work after the deliver. Teams want to take this step before completing the deliver operation because they have the opportunity to cancel rather than complete.

However, our changes look good, so we can complete this deliver operation. Because our UCM project is integrated with Rational ClearQuest, we are automatically transitioning the RCQ defect to the next stage of the workflow. Teams can choose to automatically transition ClearQuest defects to 'complete' states on a per UCM project basis.

## Optional: Showing a version tree

You may want to show a version tree of the `Readme.html` file. By taking a look at the version tree of the file, we can get a graphical picture of what happens during the deliver process. Deliver automatically takes private changes from a developer's workspace and merges them to another stream. Deliver can be done between developers, to task streams, to a common integration point, or even to teams working on different projects.

To view the version tree for this file, do the following:

1. Highlight `Readme.html`
2. Right-click it and select **Tools > Show Version Tree**.

**Figure 40. Showing the version tree**

**Figure 41. Version tree for Readme.html**

---

## Section 4. Unified Change Management: Project integrator role

At this point, you need to test the new change and create a new baseline that includes any newly delivered activities. UCM baselines are meant to represent stable configurations in your software projects. After reporting that testing has been completed, you will want to identify or label this release to ensure that you can reproduce this checkpoint at any time. Other work such as manufacturing, product support and new product development activities can then start with this new baseline.

You may be familiar with a software development organization where there is one well-identified product with multiple releases already supported, and new releases every six months. Now, imagine yourself in an Information Technology environment. Managing technology is a moving target. You and your team may produce many mission-critical custom products, often with no more than one release at a time. In either scenario, project management and managing software assets are critical to any project's success.

Rational ClearCase and Rational ClearQuest work together to keep the team focused and informed and to keep project progress moving efficiently.

Moving from the Developer environment, you'll change hats and login to Rational ClearQuest as the project *integrator*. For this demonstration, the project integrator has the combined roles of Quality Engineer and Release or Integration Engineer.

After the development work is completed, the change request proceeds to Quality Engineering to verify the change implemented and to begin testing. You, the Quality Engineer, will look up the appropriate tasks from the ClearQuest native client.

In this section you will learn how to:

- Create a baseline from the latest delivered activities

- Review, test and validate the latest changes to the activity
- Create and promote a new baseline based on your results

In this exercise the Integration Engineer will login to Rational ClearQuest as username *dana*.

1. Open the ClearQuest perspective.
2. Click on the connection icon and select **Manage Connections**.

#### Figure 42. Manage connections menu command

3. Highlight the **CQ\_Trial** connection and click **Add Connection**, and click **Next**.

#### Figure 43. ClearQuest Connection Management dialog box

4. In the Additional Connection Information dialog, under **Used ID**, enter *dana* and click **Finish**.
5. In the Connect dialog, under **Password**, enter *dana* as then and click **OK**.

#### Figure 44. ClearQuest Connect dialog box

6. Click **Close** on the ClearQuest Connection Management window. (Figure 43)

Now it's time to create your query:

1. Click on the ClearQuest menu and select **New > Query**.
2. Use the New Query Wizard to build a query that searches on **Defects** in the **Resolved** state. Name the query *Resolved Defects*. Be sure to select **Run Query** when finished.

#### Figure 45. Filtering by state

#### Figure 46. Defining the values for the filter

#### Figure 47. Defining the display fields

Refer to Steps 4 - 9 in the previous section entitled "Log on and run a query" for a reminder on how to create a query in Rational ClearQuest. This time however, you will need to include the **State** field and you will filter on the value **Resolved**.

3. Open defect **CLSLIC00000058, Fix error in readme.html** from the query results pane.

### Figure 48. Query results

4. Under **ClearQuest Record Details**, select the **Unified Change Management** tab.

### Figure 49. ClearQuest record details

5. Click the **View Change Set** button.
6. Expand the Name column on the **Change Set** tab to confirm the file name of the file, right-click `Readme.html` and select **Compare with Change Set Predecessor** from the pop-up menu.

### Figure 50. Comparing the previous version with the latest one

7. Click the Next Difference button (arrow) on the toolbar until you have viewed all of the changes to the file, and click **File > Exit**.

### Figure 51. Reviewing the changes

8. Close the Change Set properties window.
9. In the ClearQuest Record Details window, click the **Audit Trail** tab for defect CLSIC00000058 and scroll to the bottom to review what has happened to this defect thus far.

### Figure 52. Validating the changes

If this were an actual release cycle, you would run through in-depth testing and manage that process. Rational ClearQuest software manages the full range of testing activities from test planning, to test execution, to the capture and analysis of test results. Test plans can be defined. Test cases can be created and associated with specific test plans.

In this case, we will do a simple desk check for validation of the defect. To do this we will review the code changes implemented to fix this defect. It is very easy to review the changes directly from the change set information that has been collected with the defect report. By right-clicking on any of the files in the change set, you can execute many common operations, like looking at the history of the file or comparing the changed version with a predecessor. In this case, you choose to compare the new version of the source file with the version that existed before you started the change. You see that the proper change has been made.

Having confirmed that the proper fix has been you completed, you need to change the defect report to indicate that the fix has been validated. To do this, you execute the Validate action against that defect report. Once the action has been completed, the defect report is transitioned to the Closed state, and the defect is now considered fixed.

10. Select **Validate** from the **Actions** menu.

### Figure 53. The Validate command

11. The Validate action requires that you sign the record before proceeding. Click on the **eSignature** tab, and under **Enter eSignature here**, and enter *dana/dana* for username/password, and click **Apply**.

### Figure 54. The eSignature tab

12. The defect record is now closed and validated, logout out of ClearQuest.

### Figure 55. The closed record

You have now completed the lifecycle for a change request through various roles, from assigning by the project lead, through opening and resolving by the developer and validating by the tester.

## Creating a new baseline

You are now ready to incorporate the latest delivered changes into a new baseline. You go to the Rational ClearCase Metadata Explorer and navigate to the CLSICS\_CD project. You see that there are several streams, including the project's Integration stream. To create a baseline with the latest activities, right-click over the Integration stream (CLSICS\_CD\_Int) and select Make Baseline.

You have the option to create either an incremental or full baseline. The incremental baseline is more efficient because it tags only the objects that have changed since the last full baseline. For this exercise, we will create an incremental baseline.

You also have the option of accepting a default name or specifying a name for the baseline. We'll change the name of this baseline to something that's a bit easier to remember.

1. Navigate to the ClearCase perspective.
2. From the **Window** menu, go to **Show View** then select **ClearCase Metadata Navigator**.

### Figure 56. Navigating the metadata

3. Navigate to the **CLSICS\_CD** project under the **CLSICS\_pvob**, then select the **CLSICS\_CD\_Int** stream.

### Figure 57. Selecting the integration stream

4. Right-click the **CLSICS\_CD\_Int** integration stream and select **Make Baseline**.

#### Figure 58. Make Baseline command

5. Type in *QA\_Rel\_15* as the name of the new baseline. Click **OK**.

#### Figure 59. Naming the new baseline

6. Click **OK** in the Baselines Successfully Created dialog box to acknowledge creation of new baseline.

#### Figure 60. Confirming the new baseline

Once the new baseline has been created, you can look at the properties for this baseline. One of the properties tracked for each baseline is the *Promotion Level*. The Promotion Level gives you a way to track the current state or quality level of that baseline's version of your application.

When you have been able to successfully build the version of the application identified by this baseline, you change the promotion level to *Built*, as follows.

1. Right-click on the stream and select **Properties**.
2. Click on the **Baselines** tab and under **Components**, select the component (VOB) **CLSICS\_comp1**.
3. Under **Baselines**, select **QA\_Rel\_15** and click **Properties**.

#### Figure 61. Properties of stream

4. On the **General** tab in the *baseline properties* window, click the pull-down menu under **Promotion level**, set the promotion level to **BUILT**, and click **OK**.
5. Close the Stream properties window.

#### Figure 62. Promoting the baseline

---

## Section 5. Unified Change Management: Release engineer role

## Recommending the latest baseline

Now that there is a new baseline, you would like to update your development stream so that you are working with the latest version of your application. This will minimize the chance that any new work you perform will conflict with activities already delivered to the Integration stream. You also want other members of your team to be up to date with the latest changes from the integration stream. You will now take on the final role of today's exercise – *Chris*, the release engineer. Before Chris builds the software to be released, he needs to be sure that his workspace is current with the latest, recommended baseline. (Refer to the section: "Join a UCM project," above.)

1. Navigate back to the ClearCase Navigator window.
2. Join the UCM project again, but login this time as *chris/chris*. You may need to de-select the checkbox **Reuse existing connection**. Be sure to create a new development view, but don't create an integration view. Use the default names.
3. Load the **CLSICS\_comp1** VOB into the newly created view.
4. Navigate to `Readme.html` and open a version tree. NOTE: Your view is selecting an older version of the file. Close the version tree.
5. Open `Readme.html`. NOTE: You still see the spelling error.
6. Navigate back to the Metadata explorer.

Now you should recommend this baseline to your developers, as the foundation for any future work within their development streams. Recommending a baseline ensures that your teams know what is the latest, most stable baseline in the project.

7. Right-click on the integration stream again and select **Recommend Baselines**.

### Figure 63. Recommending the baseline

8. Under **Filter Promotion Level**, select **BUILT** from the pull-down menu and click the **Fetch Baselines** button.

### Figure 64. Fetching the baseline

9. Ensure that **QA\_Rel\_15** appears under the column **Baseline to**

**Recommend**, and click **OK**.

10. Close the Metadata Explorer windows.

Now that we have a "built" baseline that is *recommended*, we want to rebase our stream to get those changes in our workspace.

Much like the deliver operation, ClearCase identifies changes in the latest baseline that are not visible in your view, and performs merges where needed.

1. Click on the new view you created for Chris. From the **ClearCase** menu, select **Rebase**.

### Figure 65. Rebasing your stream

2. Click the **Advanced** button and confirm that you are rebasing the newly recommended baseline. Click **OK**.

### Figure 66. Confirming the recommended baseline

3. Login to ClearQuest as username/password *chris/chris*.
4. Click **OK** to acknowledge the rebase.
5. Click **Complete Rebase**.
6. Close the rebase confirmation window.

In this case, all of these merges are completed automatically and you choose to Complete the Rebase operation. You now have a stable configuration in your workspace that you can build and release to production.

7. Double-click `Readme.html` and verify the correction. Note that the changes made during the merge to the integration stream are now visible in your private development stream.
8. Now open a version tree and review the flow of the changes made.

## Conclusion

By now, you have seen how Rational ClearCase and Rational ClearQuest can benefit different members of your software development team.

The Project Lead:

- Is able to organize and manage the project
- Can accurately assess project status
- Can intelligently evaluate resources and assign change requests

#### The Developer:

- Can easily focus on assignments
- Can quickly create isolated environments in which to make changes
- Can deliver changes and makes those changes automatically available to the team

#### The Integrator:

- Can easily focus on assignments using custom queries
- Can easily update project status
- Can quickly create baselines and track the promotion level of those baselines

# Resources

## Learn

- This tutorial is intended to serve as a guide for an [online demonstration](#) of Unified Change management with IBM Rational ClearCase and IBM Rational ClearQuest.
- Part 2 of this series, [IBM Rational ClearCase and IBM Rational ClearQuest Evaluation Guide, Part 2: ClearQuest](#) is a tutorial that is intended to serve as a guide for an [online demonstration](#) of the IBM Rational change and configuration management solution.
- See also [Part 3](#) of this series, a tutorial intended to guide you through an [online demonstration](#) of IBM Rational ClearCase.
- To learn more about IBM Rational products, visit the [developerWorks Rational zone](#). You'll find technical documentation, how-to articles, education, downloads, product information, and more.
- Find more resources for ClearCase users and administrators in the [ClearCase area](#) of the developerWorks Rational zone, including articles and whitepapers, plug-ins, scripts and triggers; and links to training, discussion forums, product documentation and support.
- ClearQuest users and administrators can find more resources in the [ClearQuest area](#) of developerWorks Rational, including ClearQuest hooks, Eclipse plug-ins, product documentation, articles and whitepapers.
- Read about the latest release of the IBM Rational Software Development Platform in the developerWorks article: [Accelerating global software delivery](#).
- Learn more about how to benefit from IBM Rational products and technologies in [The Rational Edge e-zine](#).
- Browse the [technology bookstore](#) for books on these and other technical topics.
- Learn about [upcoming events](#); including webcasts, seminars, trade shows, user group meetings, and the IBM Rational Software Development User Conference.
- Product manuals, installation guides, and other documentation are available in the [IBM Rational Online Documentation Center](#).
- Whitepapers, analyst reports, and datasheets for IBM Rational ClearQuest are available [here](#).
- IBM Rational software has helped thousands of companies worldwide achieve success in their software development efforts. Read about some of them [here](#).

## Discuss

- The [ClearCase discussion forum](#) on developerWorks is a great place to post questions and get answers about configuration management and UCM with IBM Rational ClearCase.
- You can also join the [Rational ClearQuest forum](#) on developerWorks to post questions and communicate with other ClearQuest users.

## About the author

Paul Boustany

Paul Boustany works for IBM Rational's marketing engineering group and specializes in change and configuration management tools.