Rational Software Corporation®

Rational® ClearCase®
Rational® ClearCase LT®

Tutorial

VERSION: 2003.06.00 AND LATER

UNIX/WINDOWS EDITION

Rational®
the software development company

support@rational.com
http://www.rational.com

Legal Notices

Copyright ©1992-2003, Rational Software Corporation. All Rights Reserved.
Version Number: 2003.06.00

This manual (the "Work") is protected under the copyright laws of the United States and/or other jurisdictions, as well as various international treaties. Any reproduction or distribution of the Work is expressly prohibited without the prior written consent of Rational Software Corporation.

The Work is furnished under a license and may be used or copied only in accordance with the terms of that license. Unless specifically allowed under the license, this manual or copies of it may not be provided or otherwise made available to any other person. No title to or ownership of the manual is transferred. Read the license agreement for complete terms.

Rational Software Corporation, Rational, Rational Suite, Rational Suite ContentStudio, Rational Apex, Rational Process Workbench, Rational Rose, Rational Summit, Rational Unified process, Rational Visual Test, AnalystStudio, ClearCase, ClearCase Attache, ClearCase MultiSite, ClearDDTS, ClearGuide, ClearQuest, PerformanceStudio, PureCoverage, Purify, Quantify, Requisite, RequisitePro, RUP, SiteCheck, SiteLoad, SoDa, TestFactory, TestFoundation, TestMate and TestStudio are registered trademarks of Rational Software Corporation in the United States and are trademarks or registered trademarks in other countries. The Rational logo, Connexis, ObjecTime, Rational Developer Network, RDN, ScriptAssure, and XDE, among others, are trademarks of Rational Software Corporation in the United States and/or in other countries. All other names are used for identification purposes only and are trademarks or registered trademarks of their respective companies.

Portions covered by U.S. Patent Nos. 5,193,180 and 5,335,344 and 5,535,329 and 5,574,898 and 5,649,200 and 5,675,802 and 5,754,760 and 5,835,701 and 6,049,666 and 6,126,329 and 6,167,534 and 6,206,584. Additional U.S. Patents and International Patents pending.

U.S. Government Restricted Rights

Licensee agrees that this software and/or documentation is delivered as "commercial computer software," a "commercial item," or as "restricted computer software," as those terms are defined in DFARS 252.227, DFARS 252.211, FAR 2.101, OR FAR 52.227, (or any successor provisions thereto), whichever is applicable. The use, duplication, and disclosure of the software and/or documentation shall be subject to the terms and conditions set forth in the applicable Rational Software Corporation license agreement as provided in DFARS 227.7202, subsection (c) of FAR 52.227-19, or FAR 52.227-14, (or any successor provisions thereto), whichever is applicable.

Warranty Disclaimer

This document and its associated software may be used as stated in the underlying license agreement. Except as explicitly stated otherwise in such license agreement, and except to the extent prohibited or limited by law from jurisdiction to jurisdiction, Rational Software Corporation expressly disclaims all other warranties, express or implied, with respect to the media and software product and its documentation, including without limitation, the warranties of merchantability, non-infringement, title or fitness for a particular purpose or arising from a course of dealing, usage or trade practice, and any warranty against interference with Licensee's quiet enjoyment of the product.

Third Party Notices, Code, Licenses, and Acknowledgements

Portions Copyright ©1992-1999, Summit Software Company. All rights reserved.

Microsoft, the Microsoft logo, Active Accessibility, Active Client, Active Desktop, Active Directory,

ActiveMovie, Active Platform, ActiveStore, ActiveSync, ActiveX, Ask Maxwell, Authenticode, AutoSum, BackOffice, the BackOffice logo, bCentral, BizTalk, Bookshelf, ClearType, CodeView, DataTips, Developer Studio, Direct3D, DirectAnimation, DirectDraw, DirectInput, DirectX, DirectXJ, DoubleSpace, DriveSpace, FrontPage, Funstone, Genuine Microsoft Products logo, IntelliEye, the IntelliEye logo, IntelliMirror, IntelliSense, J/Direct, JScript, LineShare, Liquid Motion, Mapbase, MapManager, MapPoint, MapVision, Microsoft Agent logo, the Microsoft eMbedded Visual Tools logo, the Microsoft Internet Explorer logo, the Microsoft Office Compatible logo, Microsoft Press, the Microsoft Press logo, Microsoft QuickBasic, MS-DOS, MSDN, NetMeeting, NetShow, the Office logo, Outlook, PhotoDraw, PivotChart, PivotTable, PowerPoint, QuickAssembler, QuickShelf, RelayOne, Rushmore, SharePoint, SourceSafe, TipWizard, V-Chat, VideoFlash, Visual Basic, the Visual Basic logo, Visual C++, Visual C#, Visual FoxPro, Visual InterDev, Visual J++, Visual SourceSafe, Visual Studio, the Visual Studio logo, Vizact, WebBot, WebPIP, Win32, Win32s, Win64, Windows, the Windows CE logo, the Windows logo, Windows NT, the Windows Start logo, and XENIX, are either trademarks or registered trademarks of Microsoft Corporation in the United States and/or in other countries.

Sun, Sun Microsystems, the Sun Logo, Ultra, AnswerBook 2, medialib, OpenBoot, Solaris, Java, Java 3D, ShowMe TV, SunForum, SunVTS, SunFDDI, StarOffice, and SunPCi, among others, are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Purify is licensed under Sun Microsystems, Inc., U.S. Patent No. 5,404,499.

Licensee shall not incorporate any GLOBEtrotter software (FLEXlm libraries and utilities) into any product or application the primary purpose of which is software license management.

BasicScript is a registered trademark of Summit Software, Inc.

Design Patterns: Elements of Reusable Object-Oriented Software, by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides. Copyright © 1995 by Addison-Wesley Publishing Company, Inc. All rights reserved.

Copyright ©1997 OpenLink Software, Inc. All rights reserved.

This software and documentation is based in part on BSD Networking Software Release 2, licensed from the Regents of the University of California. We acknowledge the role of the Computer Systems Research Group and the Electrical Engineering and Computer Sciences Department of the University of California at Berkeley and the Other Contributors in its development.

This product includes software developed by Greg Stein <gstein@lyra.org> for use in the mod_dav module for Apache (http://www.webdav.org/mod_dav/).

Additional legal notices are described in the legal_information.html file that is included in your Rational software installation.

Welcome 1

Roadmap for Project Managers 3

Roadmap for Developers 5

Introduction. 7

Tutorial Requirements 7
Exercise Requirements 8
Starting the Tutorial 9
Location of the Tutorial Files 9
Printing the Tutorial 10

Documentation and Learning Resources 11

Accessing the Online Printed Documentation 11
Rational Learning Resources 13

Workflow in ClearCase Projects 15

Base ClearCase and UCM 15
The UCM Process 16
The Base ClearCase Process 17

Working in the ClearCase Explorer 21

Starting the ClearCase Explorer 21
Understanding the ClearCase Explorer Window 22
Customizing the ClearCase Explorer 23
Understanding the Shortcut Pane 25

Working at the Command Line 27

The cleartool Command Modes 27
About cleartool apropos (UNIX Only) 28
Accessing the Reference Pages 29
The intro Reference Page 30
Searching for Information in the Reference Pages 31

Creating VOBs 33

Types of VOBs 33

Creating a VOB Using the VOB Creation Wizard (Windows) 34
Creating a VOB on UNIX 41

Setting Up a UCM Project 43
Creating the Project Repository (PVOB) 44
Creating a Component for Storing the Project Baseline 50
Creating Components for Storing Elements 53
Creating the UCM Project 58
Creating an Integration View 72
Creating and Setting UCM Activities (UNIX Only) 77
Creating a Directory Structure and Adding to Source Control 78
Defining an Integration Baseline (Windows and UNIX) 81

Joining a UCM Project 83
About the UCM Developer Tasks 83
Joining a UCM Project 84

Mounting and Unmounting VOBs 99
Mounting VOBs 99
Unmounting VOBs 100

Working with Views 103
About Views and View Directories 103
Naming Your Views 104
Types of Views 104
Which View to Create 105
Preparing for Creating a View 106
Creating a View 107
Starting a View 115
Stopping a View 116
Updating a Snapshot View 117
Listing Active Views 118
Removing Views 119

Controlling the Contents of Views 121
How a config spec Works 121
Understanding the Role of the config spec 123

Understanding the Default config spec 124
Understanding Include Files 124
Changing the config spec 125

Checking Files In and Out 127
Checkin and Checkout Requirements 127
Adding Files to Source Control 128
About Reserved and Unreserved Checkouts 130
Checking Out Files and Directories 131
Finding Checked Out Files 133
Canceling a Checkout 136
Displaying a History of Checkouts 137
About Hijacked Files 139
When to Check In Files 139
Checking In Files 140

Working with Baselines 141
What Is a Baseline? 142
Types of Baselines 142
Comparing the Contents of Two Baselines 142
Creating a Baseline 143
Default Promotion Levels 145
Promoting a Baseline 145
Recommending a Baseline 146

Working with Branches 147
What Is a Branch? 147
When to Use Branches 148
The Config Spec-Views Connection 148
Creating a Branch 150

Delivering Your Work 153
Overview of the Deliver Process 153
Performing a Deliver Operation 154

Merging Your Work 159
Starting the Merge Manager 159

Finding Files to Be Merged 160
How ClearCase Merges 164
About Trivial and Nontrivial Merges 165

Building Software 167
About clearmake 167
About omake 168

Rebasing Your Private Work Area 169
Overview of the Rebase Process 169
Performing a Rebase Operation 170
Undoing a Rebase Operation (Windows and UNIX) 176

Index 177

Welcome 1

Welcome to the Rational ClearCase tutorial. This tutorial will show you how to start working with ClearCase and ClearCase LT.

To simplify referring to these two products, the term ClearCase is used for both ClearCase and ClearCase LT. Any product and platform differences, when applicable to the tutorial, are specified.

Tutorial Organization

The tutorial is organized in independent modules. The modular structure of the tutorial enables you to study when it is convenient to you. The tutorial does not use any scripts to set up a "tutorial" environment.

All tutorial modules identify the following:

- **The target audiences**: Project managers and developers
- **The process**: UCM and base ClearCase
- **The platforms**: Windows and UNIX
- **The products**: ClearCase and ClearCase LT

Where possible, exercises are included in the modules that enable you to practice what you learned. Some of the exercises rely on the result of the exercises in previous modules. For example, joining a UCM project to create a developer's workspace, requires that a VOB and a UCM project VOB be created by doing the exercises in the modules about creating VOBs and setting up a UCM project.

Doing the exercises is optional.

Starting the Tutorial (Windows and UNIX)

To start the ClearCase tutorial, do one of the following:

Windows

- From Start menu:

  Start > Programs > Rational Software > Rational ClearCase > Tutorial

  Start > Programs > Rational Software > Rational ClearCase LT > Tutorial

UNIX

- From the command line, type this command:

  cleartool man tutorial

Tutorial Roadmaps

Tutorial roadmaps for project managers and developers identify the modules that focus on the needs of each type of audience. However, we encourage project managers to learn about ClearCase functionality of interest to developers and vice versa.

If you are a project manager, go to the Roadmap for Project Managers.

If you are a developer, go to the Roadmap for Developers.

Roadmap for Project Managers

2

Audience Project managers

Platforms Windows, UNIX

Process UCM, base ClearCase

Products ClearCase, ClearCase LT

This roadmap identifies the modules that address the needs of project managers.

As used in the context of Rational ClearCase, project managers are individuals who manage a team of developers, content providers, and other individual contributors who all use ClearCase.

The tutorial roadmap has two tracks:

- The UCM track identifies the modules to read if you are using the UCM process.
- The base ClearCase track identifies the modules to read if you are using base ClearCase.

We suggest that you print a copy of this page and use it as a means to keep track of the modules you have completed.

Module	**UCM**	**Base ClearCase**	**Check when completed**
Introduction	x	x	
Documentation and Learning Resources	x	x	
Workflow in ClearCase Projects	x	x	
Working in the ClearCase Explorer	x	x	
Working at the Command Line	x	x	
Creating VOBs		x	

Module	UCM	Base ClearCase	Check when completed
Setting Up a UCM Project	x		
Mounting and Unmounting VOBs	x	x	
Working with Views	x	x	
Controlling the Contents of Views		x	
Checking Files In and Out	x	x	
Working with Baselines	x		
Working with Branches		x	

Roadmap for Developers

3

Audience Developers

Platforms Windows, UNIX

Process UCM, base ClearCase

Products ClearCase, ClearCase LT

This roadmap identifies the modules of the tutorial that address the needs of developers.

As used in the context of Rational ClearCase, developers are individuals such as software engineers, technical writers, and other content providers who use ClearCase in their daily work environment.

The tutorial roadmap for developers has two tracks:

- The UCM track identifies the modules to read if your development team is using the UCM process.
- The base ClearCase track identifies the modules to read if your team is using base ClearCase.

We suggest that you print a copy of this page and use it as a means to keep track of the modules you have completed.

Module	UCM	Base ClearCase	Check when completed
Introduction	x	x	
Documentation and Learning Resources	x	x	
Workflow in ClearCase Projects	x	x	
Working in the ClearCase Explorer	x	x	

Module	UCM	Base ClearCase	Check when completed
Working at the Command Line	x	x	
Creating VOBs		x	
Joining a UCM Project	x		
Mounting and Unmounting VOBs	x	x	
Working with Views	x	x	
Controlling the Contents of Views		x	
Checking Files In and Out	x	x	
Working with Baselines	x		
Working with Branches		x	
Delivering Your Work	x		
Merging Your Work	x	x	
Building Software		x	
Rebasing Your Private Work Area	x		

Introduction 4

Audience Project managers, developers

Platforms Windows, UNIX

Process UCM, base ClearCase

Products ClearCase, ClearCase LT

This module covers the following topics:

- Tutorial Requirements
- Exercise Requirements
- Starting the Tutorial
- Location of the Tutorial Files
- Printing the Tutorial

Tutorial Requirements

Before you start the tutorial, take a few moments to ensure that your system meets the following requirements:

1. Rational ClearCase or ClearCase LT is installed on your computer as a client or server.
2. Your computer has one of the following Web browsers installed:

Windows	UNIX
Internet Explorer 5.5 with Service Pack 1 or 2 Internet Explorer 6 with Service Pack 1 Netscape 4.72, 4.78, or 6.2.1	Netscape 4.72, 4.78, or 6.2.1

3. Acrobat Reader 4.0 or later is installed to view the PDF format of the tutorial.

Exercise Requirements

Exercise Requirements for ClearCase

When working on Windows, we strongly recommend that the VOBs you create during the hands-on exercises be stored locally on your computer.

To this end, you should create a dedicated tutorial folder titled **cc-tut**. The **cc-tut** folder should be a shared folder.

On UNIX, there is no need for a dedicated tutorial folder. You create the VOB in the /tmp directory.

Creating a Tutorial Folder on Windows

1. Click **My Computer** and then click **Local Disk**.
2. Click **File > New Folder**, and type **cc-tut** as folder name.
3. Right-click **cc-tut** and select **Sharing**.
4. Click **Share this folder.**
5. As a comment, type: **ClearCase tutorial exercise folder**, and click **OK**.

Exercise Requirements for ClearCase LT

When using ClearCase LT, your ClearCase administrator sets up the initial environment that allows you to create VOBs and views. This section provides you with instructions on creating such an environment using the Getting Started Wizard should it not be in place when you start the tutorial.

1. Click **Start > Programs > Rational Software > Rational ClearCase > Administration > Getting Started Wizard**.
2. On the first page of the wizard, click **Next**.
3. On the second page, accept the default **C:\ClearCaseStorage** as the VOB storage directory, and click **Next**.
4. Replace the default value (**sources**) with **cclt_tutorial_sources** as the name for the source VOB, and click **Next**.
5. Replace the default name of the initial UCM project (**InitialProject**) with **cclt_ucm_pvob**, and click **Next**.
6. The Configuration Summary dialog box indicates the storage location and project to be created:

- **Storage: C:\ClearCaseStorage**
- **CCLT Project VOB: \Projects**
- **CCLT Source VOB: cclt_tutorial_sources**
- **Initial CCLT Project: cclt_ucm_pvob**

Click **Next**.

7 ClearCase LT creates the storage environment and the various VOBs.

8 Click **Close** to close the dialog box.

Starting the Tutorial

To start the ClearCase tutorial, do one of the following:

Windows

- From the Start menu, click:

  Start > Programs > Rational Software > Rational ClearCase > Tutorial

  Start > Programs > Rational Software > Rational ClearCase LT > Tutorial

UNIX

- From the command line, type this command:

  cleartool man tutorial

Location of the Tutorial Files

The tutorial is available in the following formats: HTML, PDF, and PostScript.

The tutorial files can be found in the following default locations:

Windows

- **HTML** C:\Program Files\Rational\ClearCase\doc\tutorial\index.htm
- **PDF** C:\Program Files\Rational\ClearCase\doc\books\cc_tutorial.pdf
- **PostScript** C:\Program Files\Rational\ClearCase\doc\books\cc_tutorial.ps

UNIX

- **HTML** /opt/rational/clearcase/doc/tutorial/index.htm
- **PDF** /opt/rational/clearcase/doc/books/cc_tutorial.pdf
- **PostScript** /opt/rational/clearcase/doc/books/cc_tutorial.ps

Related Topics

Chapter 5, *Accessing the Online Printed Documentation*

Printing the Tutorial

To print the tutorial, use the print command in either your Web browser or Acrobat Reader.

Documentation and Learning Resources

5

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

As you start using Rational ClearCase, it will be helpful to know that, in addition to extensive Help, the entire printed and online documentation sets are available.

If you need more information than the documentation set provides, Rational University can help you acquire the knowledge and skills you need.

This module covers the following topics:

- Accessing the Online Printed Documentation
- Rational Learning Resources

Accessing the Online Printed Documentation

The complete online documentation set is accessible in HTML, PDF, and PostScript formats.

HTML

On Windows

From the Start menu:

Start > Programs > Rational Software > Rational ClearCase > Online Help

Start > Programs > Rational Software > Rational ClearCase LT > Online Help

From the **Help** menu in any ClearCase tool window:

Help > Help Topics. Then click **Viewing Rational ClearCase Manuals Online**.

Help > Help Topics. Then click **Viewing Rational ClearCase LT Manuals Online**.

On UNIX

From your Web browser:

1 In the **Location** field of your Web browser, type:

/opt/rational/clearcase/doc/help/index.htm

2 Click **Viewing Rational ClearCase Manuals Online** or **Viewing Rational ClearCase LT Manuals Online**

From the **Help** menu in any ClearCase tool window:

Help > Help Topics. Then click **Viewing Rational ClearCase Manuals Online**

Help > Help Topics. Then click **Viewing Rational ClearCase LT Manuals Online**

PDF

On Windows

1 From **My Computer**, navigate to **Program Files > Rational > ClearCase > Doc > Books**

2 Open the desired ClearCase manual in PDF format.

On UNIX

1 Change directory to **/opt/rational/clearcase/doc/books**

2 Start Acrobat Reader by typing: **acroread &**

3 Click **File > Open**, and click the desired book name.

Introduction	cc_intro.pdf
Managing Software Projects	cc_proj.pdf
Developing Software	cc_dev.pdf
Reference Manual (A - L)	cc_ref_1.pdf
Reference Manual (M - Z)	cc_ref_2.pdf

Try It!

1 Access the online help from the Start menu:

Start > Programs > Rational Software > Rational ClearCase > Online Help

2. Then click **Viewing Rational ClearCase Manuals Online** or **Viewing Rational ClearCase LT Manuals Online**
3. Browse one or more manuals of your choice.
4. If you are a project manager, we recommend browsing the following manuals:
   - *Introduction*
   - *Managing Software Projects*
   - *Command Reference*
5. If you are a developer, we recommend browsing the following manuals:
   - *Introduction*
   - *Developing Software*
   - *Command Reference*

Rational Learning Resources

Rational University is the central point for learning about the training resources available to you.

To find which training is available, go to the Rational University Web site: http://www.rational.com/university/index.jsp.

From there, you can obtain the following information:

- Complete course catalog
- Course descriptions
- Recommended curricula
- Training options

Try It!

1. Go the Rational Web site: http://www.rational.com.
2. Click Rational University.
3. Browse through the course catalog.
4. Browse the recommended curricula and the training options.

Workflow in ClearCase Projects

6

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

As a project manager, one of the first decisions you need to make is which process you will use to manage your projects. Should you adopt the out-of-the-box Unified Change Management (UCM) process? Or would base ClearCase be more appropriate for your organization?

This module explores the differences between the two approaches.

If you are a developer, understanding the workflow in a project using UCM or base ClearCase provides you with insight into the responsibilities of project managers and developers.

This module covers the following topics:

- Base ClearCase and UCM.
- The UCM Process.
- The Base ClearCase Process.

Base ClearCase and UCM

Base ClearCase consists of a collection of tools to establish an environment in which developers can work in parallel on a shared set of files, and project managers can define policies that govern how developers work together.

UCM is an out-of-the-box implementation of a development process layered upon base ClearCase. Adopting UCM makes it possible to manage your projects with ClearCase more quickly in an efficient manner without needing to master the details of base ClearCase.

UCM offers the convenience of a readily available management process; base ClearCase offers the flexibility to implement virtually any configuration management process that you deem appropriate for your environment.

The following sections explore the work process when using UCM and base ClearCase.

The UCM Process

When you decide to adopt UCM as the process for your development effort, as a project manager you are in charge of setting up the environment, referred to as the *UCM project*, and maintaining the shared work areas.

The UCM project is put in place after creating the VOBs that contain the files and directories that make up the development effort.

A UCM project contains one shared work area and typically multiple private work areas.

The shared work area is the data repository, called the VOB, that holds the project's source files. Private work areas allow developers to work in isolation.

When developers are ready to submit their work, they deliver the sources from their private work area to the shared work area. This action makes their contribution to the project accessible to other developers. As a project manager you are responsible for maintaining the project's shared work area.

In a UCM project, work typically progresses as follows:

1 **ClearCase Administrators create the VOBs.**

   ClearCase Administrators create the VOBs that will contain all the files and directories that represent the development effort, such as a new product release.

2 **Project managers create the UCM project**.

   As project manager, you create a UCM project and identify an initial set of components as a starting point, called a *baseline*. A component is a group of related directory and file elements, which are developed, integrated, and released together. A baseline represents a version of one or more components.

3 **Developers join the UCM project.**

   Developers join the project by creating their private work areas and populating them with the contents of the project's baselines.

4 **Development is ongoing.**

Developers create, or are assigned, activities, and work on one activity at a time. An activity records a set of files that a developer creates or modifies to complete a development task, such as fixing a bug. The set of files associated with an activity is known as a change set.

5 **Developers deliver work to the shared work area**.

When developers complete activities, they build and test their work in their private work areas. When developers create a successful software build in their private work area, they share their work with the project team by performing deliver operations. A deliver operation merges the work from the developer's private work area to the project's shared work area.

6 **Release Engineering builds the product.**

Periodically, the delivered work from developers is integrated by building the project's executable files in the shared work area. This work is typically done by an an individual in the development group who is assigned to build the product.

7 **Project managers create new baselines.**

If the project builds successfully, as a project manager you create a new set of baselines. In a separate work area, a team of Quality Engineers performs more extensive testing of the new baselines.

8 **Project managers promote specific baselines that reflect a particular project milestone.**

Periodically, as the quality and stability of baselines improve, as a project manager you adjust the promotion level attribute of the baselines to reflect appropriate milestones, such as Built, Tested, or Released. When the new baselines pass a sufficient level of testing, you designate them as the recommended set of baselines

9 **Developers adjust their private work areas to the latest available baselines.**

Developers perform rebase operations to update their private work areas to include the set of versions represented by the new recommended baselines.

Developers continue the cycle of working on activities, delivering completed activities, and updating their private work areas with new baselines.

The Base ClearCase Process

When you use base ClearCase, the flow of project management tasks follows a path that is similar to that of using UCM to manage a project.

1 **Project managers set up the project environment.**

Setting up the project environment involves the following tasks:

a Create and populate the VOBs.

As a project manager, you are responsible for creating and populating the databases, called VOBs, with the initial collection of file and directory elements.

b Define the branching strategy.

The branching policy is influenced by the development objectives of your project and provides a mechanism to control the evolution of the code base.

A branch represents a linear sequence of versions of one element. Every element has one main branch. In other words, elements contain branches; branches contain versions.

Branches may have one or more subbranches. A subbranch can be used for different lines of development. For example, you can decide to use the main branch for new development work, and a subbranch for defect fixing. Subbranches can also have subbranches.

You also establish naming conventions for branch names to facilitate identifying the type of changes the branches contain. For example, the name **rel2.1_main** identifies the branches that contain all the code for release 2.1 of your product.

A typical branching strategy is to create an integration branch and multiple development branches. Developers do their work on development branches and then merge their changes to the integration branch so other team members can see them.

Development branches can be per-developer or per-task.

c Create shared views and standard config specs.

As a project manager you will want to control how branches are created when developers check out files. To do this, you create a set of rules for developers. This set of rules is referred to as a config spec.

You also create a view that developers will share. It is a good strategy to provide an integration view for developers to use when they check in work that has evolved in isolation on a private branch.

When using ClearCase on Windows, you may decide to use the View Profiles mechanism to configure views that your project team will use. Using View Profiles promotes a specific model for the effective use of ClearCase.

d Make recommendations for view names.

You may want to establish naming conventions for views to make it easier to associate a view with the task it is used for. Although the GUI-based

view-creation tools suggest appropriate names, you may want to use something different. For example, you can require that all view names, called view tags, include the owner's username and the task, such as **bsmith_v4.0_bugfix**.

2 **Project managers implement development policies**.

To enforce development policies, you can create *metadata* to preserve information about the status of the versions, such as labels, attributes, hyperlinks, triggers and locks.

- **Define Labels**.

  A label is a user-defined name that can be attached to a version. Using labels is a powerful mechanism. For example, by applying labels you can define and preserve the relationship of a set of file and directory versions at a given point in time in the development lifecycle.

- **Define Attributes**.

  Attributes are name/value pairs that let you capture information about the state of a version from various perspectives. For example, you can attach an attribute name **CommentDensity** to each version of a source file to indicate how well the code is commented, and each attribute can have the value of **unacceptable**, **low**, **medium**, or **high**.

- **Use Hyperlinks**.

  Hyperlinks let you identify and preserve relationships between elements in one or more VOBs. For example, hyperlinks can be used to link a source file to a requirements document.

- **Use Triggers**.

  Triggers let you control the behavior of **cleartool** commands and ClearCase operations by running a specific program or executable script before or after the command runs.

- **Use Locks**.

  A lock on an object prevents all developers from modifying it. Locks are useful for implementing temporary restrictions, for example, during an integration period. Locks can also be used to retire objects, such as metadata, elements and VOBs, that are no longer used.

3 **Define global types**.

This facility makes it easy to ensure that the branch, label, attribute, hyperlink, and element types the global types need are present in all the VOBs your project uses.

4 **Establish a merging policy**.

During the lifetime of a project, the contents of individual elements diverge as they are branched and usually converge in a merge operation. As a project manager you must establish merge policies for your project.

A typical policy is to require developers to merge from the integration branch to their development branch before merging their work into the integration branch.

5 **Development is ongoing**.

Developers do their work on development branches so that their changes do not affect other developers.

6 **Developers merge work to the integration branch**.

When developers complete their work, they build and test their work on their development branch. When their work builds successfully on their development branch, they share their work with the project team by merging it to the integration branch.

7 **Release Engineering builds the product.**

Periodically, the delivered work from developers is integrated by building the project's executable files in the integration branch. This work is typically done by an individual in the development group who is assigned to build the product.

8 **Project managers labels the source files**.

project managers label the source files that went into the successful build.

9 **Developers merge the labeled sources to their development branches**.

Developers merge from the integration branch to their development branches, or adjust their config spec to pick up the labeled versions on the integration branch.

Working in the ClearCase Explorer

7

Audience	Project managers, developers
Platforms	Windows only
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

The ClearCase Explorer is the user interface that provides access to the contents of the data repositories by means of dynamic and snapshot views. When working in the ClearCase Explorer, you can define your views; add new files to source control, check out files to work on them and then check them back in; deliver your work, rebase your work area, and more.

This module covers the following topics:

- Starting the ClearCase Explorer
- Understanding the ClearCase Explorer Window
- Customizing the ClearCase Explorer
- Understanding the Shortcut Pane

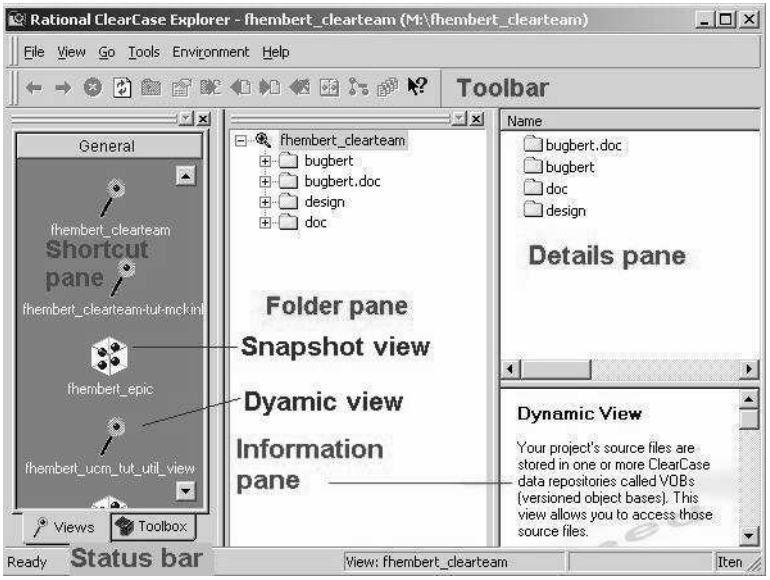
Starting the ClearCase Explorer

The quickest way to start the ClearCase Explorer is by clicking its desktop icon.

Try It!

Start the ClearCase Explorer using one of the preceding methods.

Understanding the ClearCase Explorer Window



The ClearCase Explorer window consists of several panes and a toolbar.

By default, all panes and the toolbar are displayed.

- The toolbar provides quick access to frequently used commands. Move your mouse over the icon to display the command.
- The shortcut pane holds icons for your views. At the bottom it has two tabs: **Views** and **Toolbox**.

  Under the view icon ClearCase lists the view tag, which is the name by which you reference a view.
- The folder pane identifies the root directory of the current view, and lists the data repositories, called VOBs, that are accessible under that view.

  Move your mouse over the root directory to display the view tag and its associated path.
- The detailspane displays the content of the item you selected in the folder pane.

  When you open the ClearCase Explorer for the first time, with no view selected, or when you click the **Getting Started** tab in the Shortcut pane, the Details pane then displays a Getting Started Help topic.

The details pane is replaced by a browser pane when it displays information in a Web browser.

- The status bar displays ClearCase messages.
- The information pane displays information about selected views and tools in the shortcut pane. The behavior of the information pane varies as follows:
  - When you first open a view or select it, the information pane displays summary information about the type of view.
  - When you select a single item in the **Details** pane, the information pane displays summary information about the item. If you select more than one item in the **Details** pane, or an item in the **Folder** pane, the display changes back to summary view information.
  - The information pane is only displayed when the details pane is also displayed. Both are replaced when you display the browser pane.

Customizing the ClearCase Explorer

You can customize the ClearCase Explorer in any of the following manners:

- Moving Window Panes
- Closing Window Panes
- Resizing Window Panes

Moving Window Panes

The window panes of the ClearCase Explorer are dockable. For example, by moving the folders pane under the shortcut, you increase the size of the details pane.

To move a window pane:

1. Press and hold the primary mouse button over the two horizontal lines at the top of the pane to be moved.
2. Drag the pane to the desired location and release the mouse button.

Try It!

Move the folders pane under the Shortcut pane.

Closing Window Panes

The display of the various ClearCase window panes is controlled by the View commands.

A check mark next to the command causes the window pane to display.

Clearing the check mark causes the window pane to close.

Some window panes can be closed by clicking the **Close** button **[X]**.

Try It!

1 Close the following:

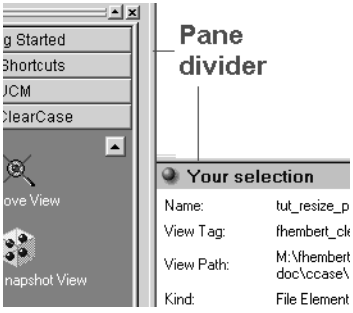
- **Information pane**
- **Folder pane**
- **Shortcut pane**
- **Toolbar**
- **Status bar**

2 Reopen the preceding window elements.

Resizing Window Panes

You can resize any of the window panes as follows:

1 Move your mouse over the pane divider until the cursor changes to resize.



2 Press the primary mouse button and drag in the desired direction.

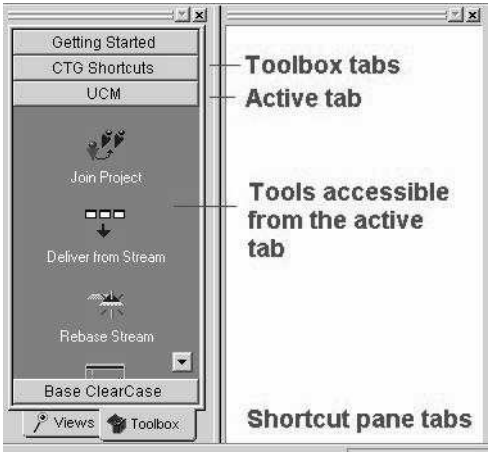
Try It!

Resize any of the following panes:

- **Information**
- **Folders**
- **Status**
- **Shortcut**

Understanding the Shortcut Pane

The Shortcut pane has two tabs at the bottom: **Views** and **Toolbox**.



Clicking the **Views** tab displays your dynamic and snapshot views.

Clicking the **Toolbox** tab gives access to another set of tabs.

ClearCase Toolbox

The default **Toolbox** tabs are the following:

- **Getting Started**. Clicking the tools icons on this tab displays information in the browser pane that provides help to get you started.
- **UCM**. This tab gives access to UCM-specific tools. From here, you can join a UCM project, deliver your work, rebase your work environment, start the Project Explorer, and so on.

Try It!

1 Click the **Toolbox** tab and then click the **Getting Started** tab and then click the tools' icon to display information about them in the information pane.

2 Click the **UCM** tab and then click the tools' icon to display information about them in the information pane.

Working at the Command Line

8

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

In addition to the graphical user interface (GUI), ClearCase functionality is accessible through a command line interface (CLI).

Using the CLI enables you to automate certain aspects of your development effort. It also gives you the choice to work using either the GUI or the CLI.

This module covers the following topics:

- The cleartool Command Modes
- About cleartool apropos (UNIX Only)
- Accessing the Reference Pages
- The intro Reference Page
- Searching for Information in the Reference Pages

The cleartool Command Modes

The **cleartool** command can be entered in two modes: single-line and interactive.

Single-Line Mode

In single-line mode, type the **cleartool** and **cleartool** subcommand on one line at the command or system prompt.

For example, these commands do the following:

- **cleartool man or cleartool man intro**

  On Windows, displays the **intro** reference page in a Web browser.

On UNIX, displays the **intro** reference page in a terminal window.

The **intro** reference page is discussed in more detail later in this module.

- **cleartool man -graphical** or **cleartool man intro -graphical**

  On UNIX, displays the intro reference page in a Web browser.

- **cleartool man** *command*

  On Windows and UNIX, opens the reference page for the specified **cleartool** command. For example: **cleartool man checkout.**

Try It!

1. Open a command window.
2. Enter the command to bring up the intro reference page, and review the contents.
3. Enter the command to bring up the reference page for the **cleartool checkin** command, and review the contents of the reference page.
4. Practice opening the intro and other reference pages in a Web browser.

Interactive Mode

In interactive mode, you first type the **cleartool** command at the command prompt. Then at the cleartool prompt, you enter the **cleartool man** subcommand followed by the command for which you want to obtain help.

To return to the system prompt, type **exit**.

For example:

1. **system-prompt> cleartool**
2. **cleartool> man**

   This opens the **intro** reference page.

   The command **cleartool > man** opens the reference page for the cleartool command itself.
3. **cleartool> exit**

About cleartool apropos (UNIX Only)

The **cleartool apropos** command provides a summary explanation of a **cleartool** command. The command format is: **cleartool apropos** *command*.

For example, when you type **cleartool apropos lsview** at the system prompt, ClearCase returns the following information: **Lists view registry entries**.

Try It!

Using cleartool apropos, display information about the following commands:

- **mkvob**
- **lsvob**
- **mklabel**
- **deliver**
- **merge.**

Compare the system response with responses returned for any of the preceding commands.

Accessing the Reference Pages

In addition to being available as printed books, the complete command line reference manual is accessible online.

To open the online reference manual on Windows and UNIX platforms, proceed as follows:

On Windows

You can access the reference manual from these locations:

- **From the Start menu:**

  Start > Programs > Rational Software > Rational ClearCase > Online Help > Viewing Rational ClearCase Manuals Online > Command Reference

  Start > Programs > Rational Software > Rational ClearCase LT > Online Help > Viewing Rational ClearCase LT Manuals Online > Command Reference

- From the **Help** menu:

  Click Help > Help Topics > Rational ClearCase Reference

The reference manual opens as HTML pages in a Web browser.

On UNIX

The reference manual is available in two formats: ASCII and HTML.

- From the command line:

cleartool man contents

This command opens the table of contents of the reference manual in a Web browser.

- **From the Help menu:**

  Click **Help > Help Topics > Rational ClearCase Reference**

Try It!

1. Using the Help menu, open the Rational ClearCase Reference manual.
2. Review the reference pages for any cleartool commands of interest to you.
3. Using the CLI, display the same reference pages using the single-line mode as ASCII and HTML pages.

The intro Reference Page

The **intro** reference page provides a general orientation to the contents of the reference manual.

The content of the **intro** reference page is organized in sections that help new and experienced users quickly find information about the most frequently used commands at the command prompt, such as:

- Commands to start tools with graphical user interfaces on UNIX
- Commands listed by user roles, namely:
  - Developers
  - Project managers
  - Administrators

Try It!

Display the **intro** reference page using these methods:

1. From the CLI, using the single-line mode: **cleartool man**
2. From the online Reference manual through the Help menu.

Searching for Information in the Reference Pages

You can search the reference manuals using the search mechanisms available on Windows and UNIX platforms.

Searching for Information on Windows

1. From **My Computer**, navigate to **Program Files > Rational Software > ClearCase > Doc > Help** and click **Search**.
2. In the **Search** field, type: **ct_ref**
3. Click **Advanced**, select **Containing text**, and type the text string to search for.

Searching for Information on UNIX

1. In a terminal window, navigate to **<CCASEHOMEDIR>/doc/help/ct_ref**/
2. At the command prompt, type one of the following:
   - **grep** *string* ***.htm**
   - **grep '***string with spaces***' *.htm**

Creating VOBs

9

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	Base ClearCase
Products	ClearCase, ClearCase LT

A VOB (versioned object database) is a data repository that holds the files, directories, and any objects that collectively represent a product or a development effort.

As the project manager, when you use UCM as the process for managing your projects, you are responsible for setting up the project environment, referred to as a *UCM project*. During the creation of the UCM project, ClearCase creates the project VOB as well as the development and integration streams. This subject is covered in the module *Setting Up a UCM Project*.

When you use base ClearCase, you have to create the VOBs manually.

This module covers the following topics:

- Types of VOBs
- Creating a VOB Using the VOB Creation Wizard (Windows)
- Creating a VOB on UNIX

Types of VOBs

Using ClearCase, you can create public and private VOBs.

Public VOBs are accessible to the development community; private VOBs are only accessible to the people who created them.

VOBs can be created using a wizard or using the CLI.

To keep the tutorial simple to follow, the following sections describe how to create a VOB using the VOB Creation Wizard on Windows, and the **mkvob** command on UNIX.

Creating a VOB Using the VOB Creation Wizard (Windows)

NOTE for ClearCase users: When using ClearCase, to do the exercise on Windows you need a shared folder named cc-tut on your computer. See the section *Exercise Requirements for ClearCase* in the module *Introduction* for instructions on creating that shared folder.

NOTE for ClearCase LT users: When using ClearCase LT, a storage directory must have been created as well as initial VOBs. See the section *Exercise Requirements for ClearCase LT* in the module *Introduction* for set-up instructions.

One method to create a VOB is to use the Create a VOB Wizard. The wizard guides you through creating the project's repository as well as creating the components for storing the project's elements.

The following sections explain how to use the VOB Creation Wizard and provide you with instructions to complete each page of the VOB Creation Wizard.

Follow the directions of the platform of interest to you:

- Creating a VOB Using ClearCase
- Creating a VOB Using ClearCase LT

Creating a VOB Using ClearCase

Starting the Create VOB Wizard

Start the wizard by clicking **Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB**.

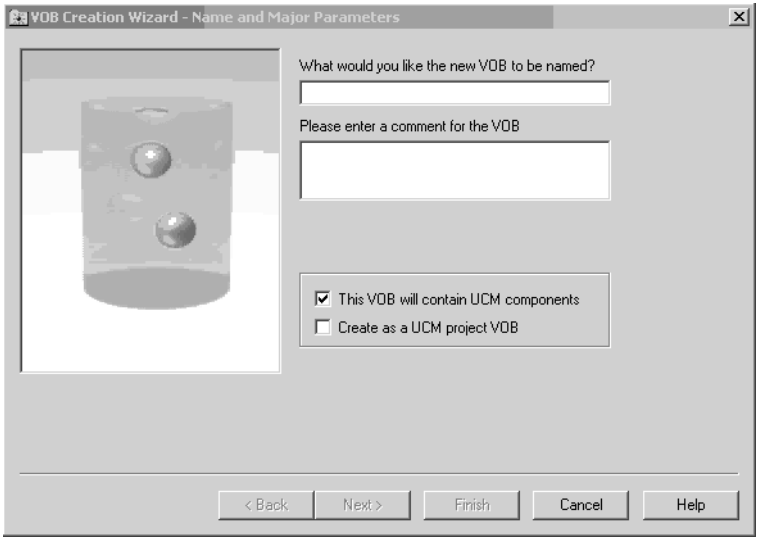
The wizard guides you through several pages where you make your selections.

Try It!

Start the VOB Creation Wizard from the Start menu.

Completing the Name and Major Parameters Page

The **Name and Parameters** page of the wizard prompts you for a VOB name andwhether components should be stored in the VOB. It also provides an option for creating a UCM project VOB.



You complete the first page of the wizard as follows:

1. Specify a name for the VOB. To facilitate identifying VOB ownership of private VOBs, it is good practice to precede the VOB name with your username. For example, jsmith_vob.
2. Enter a comment to describe the purpose of the VOB.
3. You can store multiple components in a VOB, or you can create a VOB that stores one component. If your project uses a small number of components, you may want to use one VOB for each component. But if your project uses many components, you may want to store multiple components in several VOBs.

   UCM only. When your team adopts UCM, and you select the option **This VOB will contain UCM components**, there will be one component stored for each VOB. When you clear this option, you are creating a VOB that can store multiple components.

   Base ClearCase. When you do not use UCM but base ClearCase, you need to clear this option.
4. Unless you are creating a UCM Project VOB, you should leave the option **Create as a UCM project VOB** cleared.

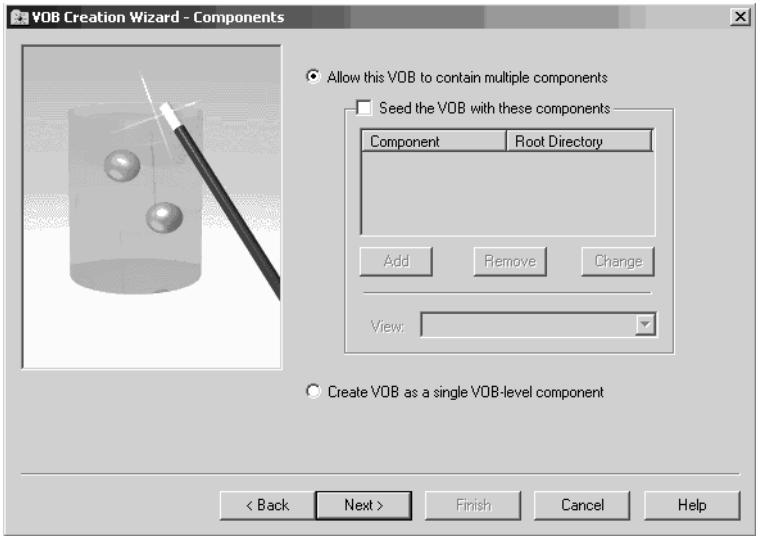
Try It!

Complete the **Name and Major Parameters** page as follows:

- **VOB name**. Type: *your-username*_**clearcase_tutorial**. *For example, jsmith_clearcase*_**tutorial.**
- **Comment**. Type: tutorial exercise VOB
- Clear the option **This VOB will contain UCM components.**

Completing the Components Page

The options on the **Components** page pertain to UCM components. The **Options** page appears only when you select the option **This VOB will contain UCM components** on the **Name and Parameters** page of the VOB Creation Wizard. If you cleared this option, the next page displayed is the **Storage** page of the VOB Creation Wizard.



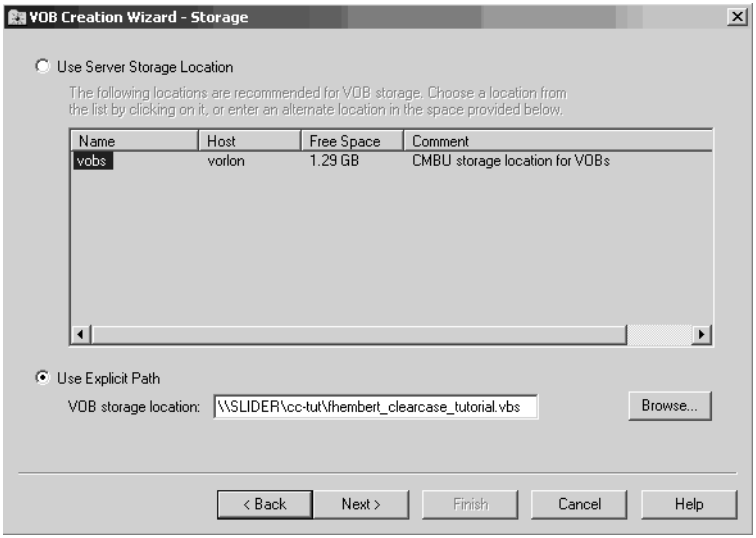
The default selection is to allow the VOB you are creating to contain multiple components.

Try It!

Since you cleared the option **This VOB will contain UCM components** on the **Name and Parameters** page of the VOB Creation Wizard, the **Components** page does not display, and the next page is the **Storage** page.

Completing the Storage Page

On the **Storage** page of the wizard, you specify the VOB's storage directory. The storage directory is a directory tree that serves as the repository for the VOB's content.



You can choose from one of the recommended locations or enter the path of a different location.

The location must be a shared resource location. The **Browse** button lets you search the network for shared resource locations.

Try It!

You are creating a VOB located on your system. Complete the **Storage** page as follows:

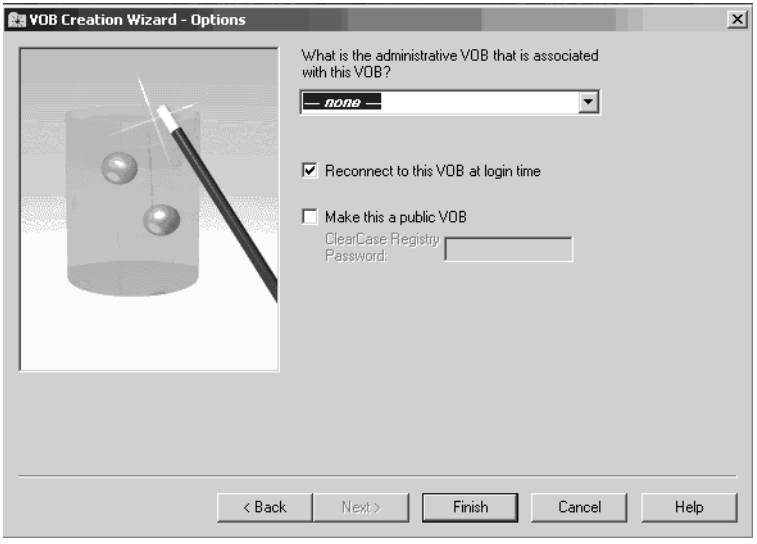
1. Select the option **Use Explicit Path** and click **Browse**.
2. Navigate to the shared cc-tut tutorial folder and double-click it. Notice ClearCase completed the name of the VOB storage directory: *your-username*_clearcase_tutorial.vbs.
3. Click **OK**.

   The VOB storage location should be similar to ***your-computer-name*\cc-tut***your-username***_clearcase_tutorial.vbs**.
4. Click **Next**.

Completing the Options Page

If you cleared the option **This VOB will contain UCM components** on the **Name and Parameters** page of the VOB Creation Wizard, you are now prompted to specify an administrative VOB to be associated with the VOB you are creating. The administrative VOB is a repository for global types of objects that VOBs can share.

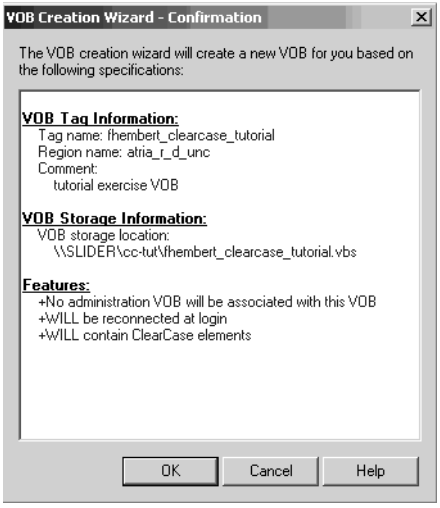
If you selected the option **This VOB will contain UCM components**, you are then prompted to select a project VOB to associate with the VOB you are creating.



When you create components, ClearCase makes AdminVOB hyperlinks between the components and the PVOB, and the PVOB assumes the role of administrative VOB.

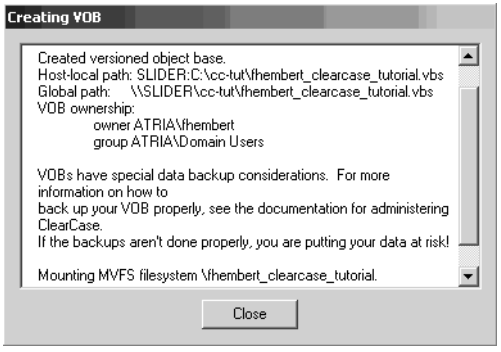
Also on this page of the wizard can you also specify if you want to create a public VOB, and reconnect at login.

After you click **Finish**, ClearCase displays a confirmation dialog box, listing your selections.



If you change your mind, you can click **Cancel** to return to the VOB Creation Wizard. Click **Previous** until you reach the page containing the options you want to change.

When you click **OK**, ClearCase creates the VOB and displays summary information in the Creating VOB message box.



Click **Close** to close the box.

Try It!

Complete the **Options** page as follows:

1. Select **None** from the list under **What is the Administrative VOB that is associated with this VOB**.
2. You are creating a private VOB, so make sure the option **Make this a public VOB** is cleared.

3 Click **Finish**.

4 ClearCase opens a confirmation dialog box which lists your VOB specifications. Click **OK** to close it.

5 When the VOB is created, ClearCase displays summary information. Any errors incurred during the VOB creation process are listed here. Notice that ClearCase automatically mounted the VOB you just created.

6 Click **Close** to close the message box.

Creating a VOB Using ClearCase LT

Start the wizard by clicking **Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB**.

The wizard guides you through several pages where you make your selections.

Try It!

Start the VOB Creation Wizard from the Start menu.

Specifying the Name and Major Parameters

In this page of the wizard, you specify the name of the VOB, or VOB tag.

Try It!

Complete the **Name and Major Parameters** page as follows:

- VOB name: *your-username*_**cclt_tutorial**. For example, **rvk_cclt_tutorial**.
- Comment: **tutorial exercise VOB**
- Click **Next**.

Specifying the Components

In this page you identify components that pertain to UCM projects. You also specify if the VOB can contain more than one component.

Try It!

1 Accept the default value: **Allow this VOB to contain multiple components**.

2 Click **Finish**.

3 On the **Confirmation** page, click **OK**.

ClearCase LT creates the VOB and mounts it automatically.

4 Click **Close**.

Creating a VOB on UNIX

To create a VOB on UNIX, you use the **cleartool mkvob** command.

The **mkvob** command does the following:

- Creates a VOB storage directory at the path you specify. The VOB storage directory contains database and a set of storage tools.
- Creates a VOB tag with which the developers access the VOB.
- Places entries in the network's VOB registry.
- Starts a VOB server process on the VOB server host.

Note that, depending on the platform and ClearCase product you use, not all command options are available. Also, if you use ClearCase LT, you must issue the command on the ClearCase LT server host.

Creating the VOB with the **mkvob** command does not automatically mount the VOB. To mount the VOB, use the **mount** command.

To create a VOB using the **mkvob** command, do the following:

- In a command window, enter this command. The options shown in this command are available on all platforms and for all ClearCase products.

  cleartool mkvob -tag *vob-tag* **-comment** *comment* **-tcomment** ***tag-comment*** **-stgloc** *vob-stgloc-name*

Let's examine what happens when you issue the command as shown with these parameters.

- **mkvob** creates a VOB. The command also creates a VOB root directory and a lost+found directory.
  - **VOB root directory.** A **mkdir** command is implicitly run to create the VOB root directory element in the new VOB. Activating a VOB makes its root directory accessible at the path specified by the VOB tag.
  - **lost+found directory**. In ClearCase, **mkvob** also creates a special directory element, lost+found, as a subdirectory of the VOB root directory. In this directory are placed elements that are no longer entered in any versioned directory.
- **-tag** *vob-tag* specifies the name and path for the VOB.

- **-comment** *comment* specifies a comment for all event records created by the command. The comment string must be a single line; typically, you must enter it between single or double quotes.
- **-tcomment** *tag-comment* adds a comment to the VOB tag's entry in the vob_tag registry file.
- **-stgloc** *vob-stgloc-name* lets you specify the name and path of the storage server.

Unless you use the **-public** option, the VOBs you create are private VOBs.

The preceding is just a sample of available options to create a VOB with the **mkvob** command.

Try It!

Type this command at the system prompt of a terminal window:

cleartool mkvob -tag /tmp/*your-username***_clearcase_tutorial -tcomment "***tutorial exercise VOB***" -comment "***initial creation of the tutorial VOB***" ~/***your-username***_clearcase_tutorial.vbs**

For example: **cleartool mkvob -tag /tmp/***jsmith***_clearcase_tutorial -tcomment "***tutorial exercise VOB***" -comment "***initial creation of the tutorial VOB***" ~/***jsmith***_clearcase_tutorial.vbs**

When successful, the system response is similar to the following:

```
Created versioned object base.
Host-local path: goose:/export/home/fhembert/fhembert_clearcase_tutorial.vbs
Global path: /net/goose/export/home/fhembert_clearcase_tutorial.vbs

VOB ownership:
owner stellar.com/fhembert
group stellar.com/user

VOBs have special data backup considerations. For more information on how to
back up your VOB properly, see the documentation for administering ClearCase.
If the backups aren't done properly, you are putting you data at risk!
```

Setting Up a UCM Project

10

Audience	Project managers
Platforms	Windows, UNIX
Process	UCM
Products	ClearCase, ClearCase LT

Unified Change Management (UCM) is an out-of-the-box project management process layered on top of base ClearCase. UCM simplifies the management of projects whose sources are put under source control.

As a project manager of a UCM-managed project, you will be setting up the project environment. This module explains the project manager's tasks to set up a UCM project.

Note for ClearCase users: To do the exercises on Windows, you need a shared folder named **cc-tut**. See *Exercise Requirements* in the module *Introduction* for steps on creating the shared folder.

Note for ClearCase LT users: You need VOB Storage location created by the ClearCase Administrator during installation. See *Exercise Requirements* in the module *Introduction* for steps on creating the shared folder.

This module covers the following topics:

- Creating the Project Repository (PVOB)
- Creating a Component for Storing the Project Baseline
- Creating Components for Storing Elements
- Creating the UCM Project
- Creating an Integration View
- Creating and Setting UCM Activities (UNIX Only)
- Creating a Directory Structure and Adding to Source Control
- Defining an Integration Baseline (Windows and UNIX)

Creating the Project Repository (PVOB)

In ClearCase, a UCM project contains the configuration information needed to manage a significant development effort. You can think of it as the development environment for a team of developers working on a product release.

This project information is stored in a dedicated repository called the *project VOB* (*PVOB)*.

The directories and the files the developers create and work on are stored separately from the PVOB in one or more data repositories called VOBs. When you set up these VOBs, you then associate the VOBs with the PVOB.

The development environment of a UCM project consists, typically, of one shared work area and multiple private work areas. Private work areas allow developers to work on activities and test their work in isolation, before submitting their work to the main shared area.

Note: This task of creating a PVOB does not apply to ClearCase LT project managers as the Administrator creates the PVOB during the installation.

Follow the directions of the platform of interest to you:

- Creating a PVOB on Windows
- Creating a PVOB on UNIX

Creating a PVOB on Windows

You use the VOB Creation Wizard to create the PVOB.

Starting the VOB Creation Wizard

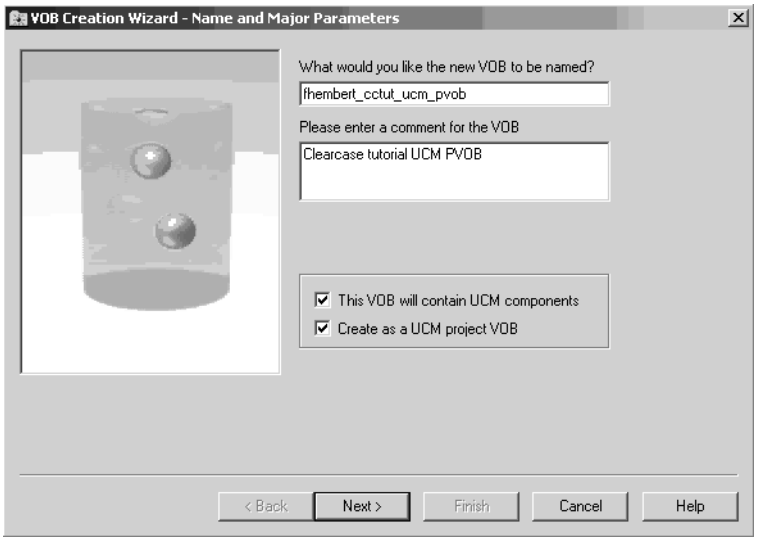
You start the wizard by clicking **Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB**.

The wizard guides you through several pages where you will make your selections.

Try It!

Start the VOB Creation Wizard.

Completing the Name and Parameters Page



The **Name and Parameters** page of the wizard prompts you for a VOB name, how components should be stored in the VOB, and if you are creating a UCM project VOB.

You complete the first page of the wizard as follows:

1. Specify a name for the PVOB. The name must be unique. We recommend that you add **_pvob** at the end of the name you specify. This makes it easy to distinguish a UCM PVOB from other VOBs.
2. Enter a comment to describe the purpose of the PVOB.
3. When you select the option **This VOB will contain UCM components**, you are creating a VOB that can store multiple components.

   We recommend against doing so unless your project is very small and you anticipate that it will remain small.
4. When creating a UCM PVOB, make sure to select the option **Create as a UCM project VOB**.

Try It!

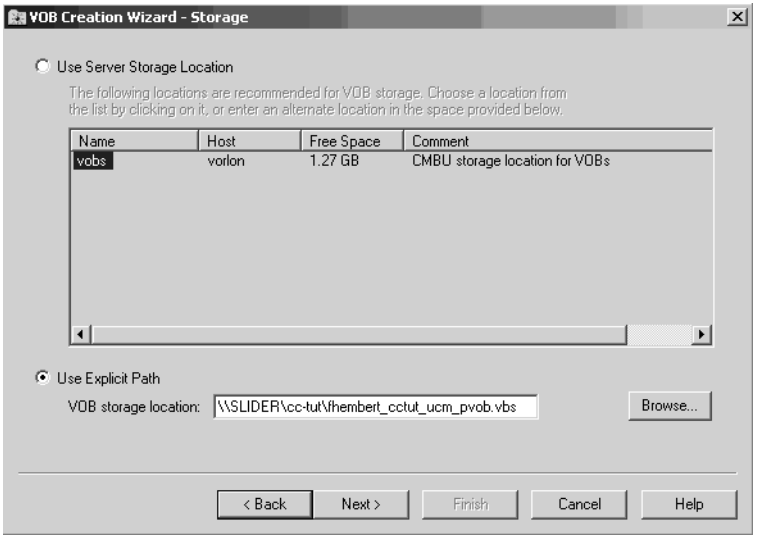
Complete the **Name and Parameters** page as follows:

- **VOB name**. Type *your-username*_**cctut_ucm_pvob**
- **Comment**. Type: **UCM tutorial PVOB**
- Make sure the option **This VOB will contain UCM components** is clear.
- Select the option **Create as a UCM project VOB**.

- Click **Next**.

Completing the Storage Page

The Storage page allows you to specify the storage directory of the PVOB. This is a directory tree that serves as the repository for the PVOB's content.



You can choose from one of the recommended locations or enter a different path using the universal naming convention.

The location must be a shared resource location. The **Browse** button lets you search for shared resource locations on the network or on your computer.

Try It!

You are creating the PVOB on your computer in the shared folder cc-tut.

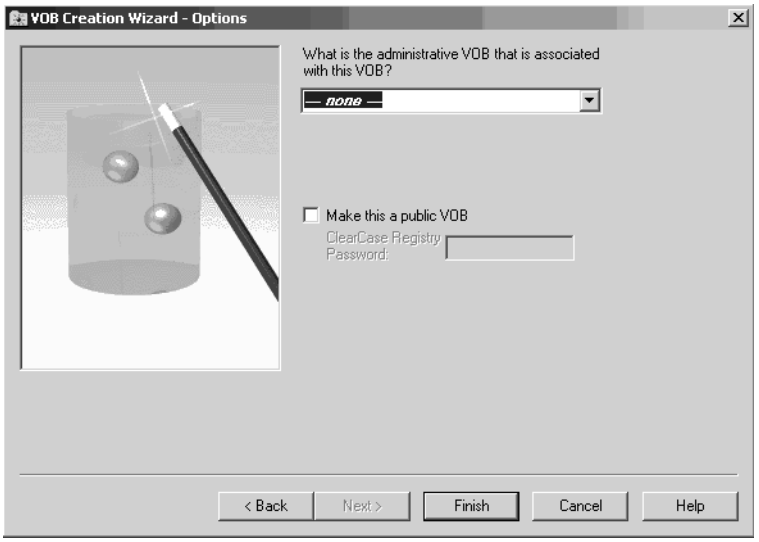
Complete the Storage page as follows:

1. Select the option **Use Explicit Path** and click **Browse**.
2. Navigate to the shared **cc-tut** folder you created, and double-click it. Notice ClearCase completed the name of the VOB storage directory: *your-username*_cctut_ucm_pvob.vbs.
3. Click **OK**. The VOB storage location on the **Storage** page of the wizard should be similar to *your-computer-name*\cc-tut*your-username*_cctut_ucm_pvob.vbs**.**
4. Click **Next**.

Completing the Options Page

Note: This page does not exist in ClearCase LT.

On the **Options** and final page of the wizard, you are prompted to specify a VOB to be associated with the PVOB you are creating. This VOB becomes the administrative VOB and is a repository for global types of objects that VOBs can share.

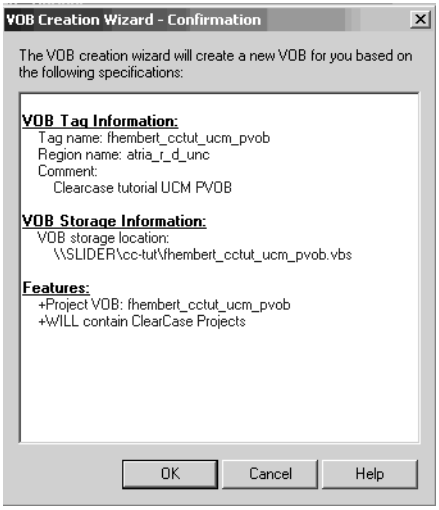


Because in UCM, the PVOB acts as administrative VOB for its components, you select **none** from the list.

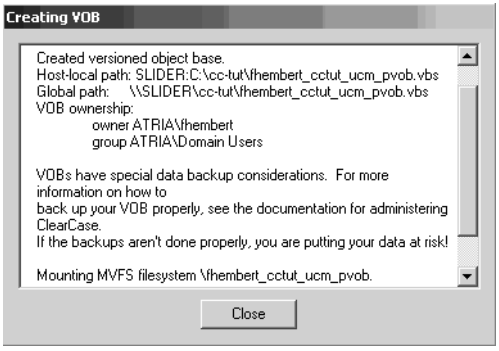
On this page of the wizard you can also specify if you want to create a public VOB. Select this option to make the VOB accessible to other individuals so that they can add files and other artifacts to source control. In the context of the tutorial, you are creating a private PVOB.

This page is the last of the wizard. Before you click the **Finish** button you can still modify any of your previous selections by moving around in the wizard's pages using the **Previous** and **Next** buttons to access the pages that contain the options you want to change. Clicking **Cancel** aborts the operation.

When you click **Finish**, ClearCase displays a Confirmation dialog box listing the selections you made.



After clicking **OK**, ClearCase starts the process of creating the PVOB, displaying status information as the creation process progresses. When done, ClearCase displays summary information in the Creating VOB dialog box.



Notice the last line indicating that ClearCase automatically mounts the PVOB.

Try It!

Complete the **Options** page as follows:

1. Select **none** from the project VOB list. In UCM, the PVOB acts as administrative Make sure the option **Make this a public VOB** is cleared. In a real development environment, you would make a public VOB to allow the project team members to access the VOB.
2. Click **Finish**.

3 ClearCase opens a confirmation dialog box which lists your VOB specifications. Click **OK** to close it.

4 ClearCase displays progress information, and when the VOB is created, ClearCase opens a summary dialog box. If any errors were incurred during the VOB creation process, they are listed here. Click **Close** to close the dialog box.

Where Am I?

You finished creating the UCM project VOB on your computer in the cc-tut shared folder.. Your next task will be that of creating a component for storing the project's baseline.

Creating a PVOB on UNIX

Note: This task does not apply to ClearCase LT users. The ClearCase Administrator creates the PVOB during the installation.

Creating a PVOB on UNIX involves the following tasks:

1 Creating the PVOB using the **mkvob** command.

2 Creating the PVOB's mount point and mounting the PVOB, when working in a MVFS environment, and developers use dynamic views.

Note: This second task is not applicable to ClearCase LT users.

The following sections describe these tasks in more detail.

1 To create a PVOB on UNIX, you use the **mkvob** command. For example:

cleartool mkvob -tag /vobs/myproj2_pvob -nc -ucmproject \ /usr/vobstore/myproj2_pvob.vbs

Let us examine the command in more detail.

The **-ucmproject indicates that you are creating a PVOB instead of a regular VOB.**

The **/vobs/myproj2_pvob tag is the name of the PVOB.**

The **-nc** option indicates that you choose not to add a comment line. To add a comment line, use the **-co** option and type the comment string between quotes.

The **/usr/vobstore/myproj2_pvob.vbs path specifies the location of the PVOB's storage directory. A PVOB storage directory is a directory tree that serves as the repository for the PVOB's content.**

2 Your next task is to create the PVOB's mount point. For example:

mkdir /vobs/myproj2_pvob.

The name of the directory you are creating must match the PVOB-tag used in the preceding step.

3 Next, you mount the PVOB using the **cleartool mount** command. For example:

cleartool mount /vobs/myproj2_pvob

Try It!

You are going to create a PVOB in the /var/tmp/ directory.

1 Create a UCM PVOB by typing this command:

cleartool mkvob -tag /var/tmp/*your-username***_cctut_ucm_pvob -co "UCM tutorial PVOB" -ucmproject /var/tmp/***your-username***_cctut_ucm_pvob.vbs**

When successful, the system response is similar to the following:

```
Created versioned object base.
Host-local path: goose:/var/tmp/fhembert_cctut_ucm_pvob.vbs
Global path: /net/goose/var/tmp/fhembert_cctut_ucm_pvob.vbs

VOB ownership:
owner stellar.com/fhembert
group stellar.com/user

VOBs have special data backup considerations. For more information on how to
back up your VOB properly, see the documentation for administering ClearCase.
If the backups aren't done properly, you are putting you data at risk!
```

Do the next two steps only if you are working in MVFS and use a dynamic view.

2 Create the PVOB's mount point using the **mkdir** command:

mkdir /var/tmp/*your-username***_cctut_ucm_pvob**

3 Mount the PVOB by typing using the **cleartool mount** command:

cleartool mount /tmp/*your-username***_tutorial_ucm_pvob**

Creating a Component for Storing the Project Baseline

Afer creating the PVOB, the next task is to create a component to store the project baseline.

Using a *composite baseline* to represent the project is easier than keeping track of a set of baselines, one for each component. This enables you to dedicate one component for storing the project baseline.

To ensure that nobody creates elements in this component, you create a component without a VOB root directory. A component that has no VOB root directory cannot store its own elements.

To create a component without a VOB root directory you use the Project Explorer. This is the graphical user interface (GUI) through which you create, manage, and view information about projects.

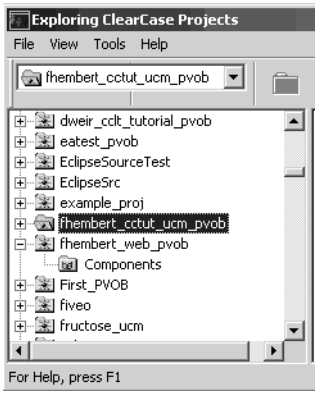
Starting the Project Explorer

The Project Explorer is available on both UNIX and Windows platforms.

You start the Project Explorer as follows:

- On Windows, by clicking **Start > Programs > Rational Software > Rational ClearCase (**or **ClearCase LT) > Project Explorer**.
- On UNIX by typing **clearprojexp &** at the system prompt in a terminal window.

Figure 1 The Project's Root Folder



The left pane of the Project Explorer lists folders for all PVOBs in the local ClearCase domain. Each PVOB has its own root folder. All VOBs shown in the example are UCM PVOBs.

In Figure 1, the selected root folder is the tutorial's PVOB: **fhembert_cctut_ucm_pvob**. ClearCase creates the root folder using the name of the PVOB.

Figure 2 The Project ‘s Component Folder

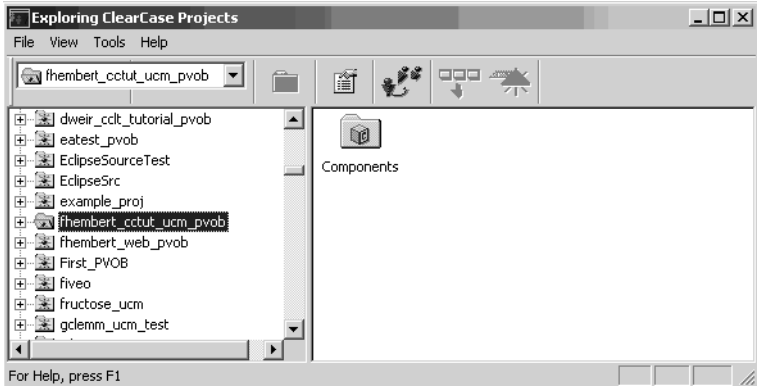


Figure 2 also shows the empty folder called Components that ClearCase created when you created the PVOB. ClearCase uses this folder to store the project components that you are going to create in the next exercise.

The following sections guide you in creating a project component baseline.

Follow the directions of the platform of interest to you:

- Creating a Project Component Baseline on Windows
- Creating a Project Component Baseline on UNIX

Creating a Project Component Baseline on Windows

You are going to create a project component named **tutorial**.

Try It!

1. Start the Project Explorer.
2. Navigate to the PVOB you created: *your-username*_**cctut_ucm_pvob**
3. Double-click the PVOB to open it and reveal the **Components** folder.
4. Right-click the **Components** folder and select the following:
   - On Windows: **New > Component without a VOB**
   - On UNIX: **New > Create Component without a VOB**.
5. In the Create Component without a VOB dialog box, type **tutorial** as component name, and click **OK**.

Creating a Project Component Baseline on UNIX

You are going to create a project component named **baselinecomp**.

Try It!

1. Start the Project Explorer by typing: **clearprojexp &**
2. Navigate to the PVOB you created: **/var/tmp/***your-username***_cctut_ucm_pvob**
3. Double-click the PVOB to reveal the **Components** folder.
4. Right-click the **Components** folder and select **New > Create Component without a VOB**.
5. In the Create Component without a VOB dialog box, do the following:
   a. Type as component name: **baselinecomp** as component
   b. Type as comment: Stores component baselines
   c. Click **OK**.

Where Am I?

So far, you have completed the following tasks:

- Created a PVOB
- Cerated a component to store the project baseline.

Your next task is that of creating a component for storing the project's elements.

Creating Components for Storing Elements

This task focuses on creating components for storing the files that developers create for the project. You use the VOB Creation Wizard to create the components.

When you create a component for storing elements, you must specify the VOB that stores the component's directory tree. You can store multiple components in a VOB, or you can create a VOB that stores one component.

In a real development environment, you would create multiple components as this makes a better use of the VOBs. The tutorial focuses on creating one component per VOB.

To create a component, follow instructions for the platform of interest to you.

- Creating a Component for Storing Elements on Windows
  - Using ClearCase

- Using ClearCase LT
- Creating a Component for Storing Elements on UNIX
  - Using ClearCase
  - Using ClearCase LT on UNIX

Creating a Component for Storing Elements on Windows

Using ClearCase

Try It!

1 Start the VOB Creation Wizard:

Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB.

The **Name and Parameters** page enables you to enter basic information such as the component's name, a comment to describe the purpose of the component, and whether the VOB you are creating will contain UCM components.

2 Complete the **Name and Parameters** page as follows:

a Name: *your-username*_**tut_elements_vob**

b Comment: **VOB to store tutorial elements**

c Select the option **This VOB will contain UCM components**.

d Click **Next.**

The **Components** page opens. On this page, you select the type of VOB and the view in which to perform the operation.

3 Complete the **Components** page as follows:

a Select the option **Create the VOB as a single VOB-level component.**

b Click **Next**.

The next page is the **Storage** page. On this page you specify where to store the component. You can choose one of the recommended locations or enter the UNC path of a different location. Click **Browse** to search the network for shared resource locations.

4 Complete the **Storage** page as follows:

a Verify that the option **Use Explicit Path** is selected.

b Click **Browse**.

c Navigate to the shared folder cc-tut.

d Double-click **cc-tut.**

e Click **OK**.

f Verify that the Explicit Path lists a storage path similar to this: *your-computer***cc-tut***your-username*_**tut_elements_vob.vbs**.

g Click **Next.**

On the **Options** page, you identify the PVOB that will store the project information about the component.

5 Complete the **Options** page as follows:

NOTE: There is no Options page in ClearCase LT.

a From the list, select the UCM PVOB you created earlier: *your-username*_**cctut_ucm_pvob**.

b Click **Finish**.

6 Click **OK** in the **Confirmation** message box.

7 Click **Close** in the **Summary** message box.

ClearCase creates the component with an initial baseline that points to the \main\0 version of the component's root directory.

Using ClearCase LT

To create a VOB and its one component in ClearCase LT:

1 Click **Start > Programs > Rational ClearCase LT Server > ClearCase Create VOB**. The VOB Creation Wizard appears.

2 In Step 1, do the following:

a Enter the name of the component: *your-username*_**tut_elements_vob**.

b Enter as comment to describe the purpose of the component: **VOB to store tutorial elements**.

1 In Step 2, select the **Create VOB as a single VOB-level component** check box. The wizard needs a view in which to perform the operation. Select a view from the **View** list.

2 In Step 3, you specify the path to the storage location for the VOB's storage directory. This page of the wizard lists the VOB storage locations created by your ClearCase administrator. If only one VOB storage location exists, the VOB Creation Wizard skips this step and uses that VOB storage location.

Creating a Component for Storing Elements on UNIX

Using ClearCase

To create a VOB and its one component in ClearCase, you must perform the following tasks:

1. Create a view using the **cleartool mkview** command.
2. Set the view using the **clearool setview** command.
3. Create a VOB using the **cleartool mkvob** command.
4. Create a mount point for the VOB using the **mkdir** command.
5. Mount the VOB using the **cleartool mount** command.
6. Creating a component using the **cleartool mkcomp** command.

The exercise guides you in accomplishing these tasks.

Try It!

You are going to create a view named *compview*, a VOB named **/var/tmp/***your-username***_elements_vob,** and associate them with the UCM PVOB **/var/tmp/***your-username***_cctut_ucm_pvob** created earlier.

Note for ClearCase LT users: Go to *Using ClearCase LT on UNIX* for instructions.

At the system prompt of a terminal window, do the following:

1. Create a view named compview by typing this command:

   cleartool mkview -tag compview /var/tmp/compview.vws

   The system's response is similar to this:

   ```
   Created view.
   Host-local path: stellar:/var/tmp/compview.vws
   Global path: /net/stellar/var/tmp/compview.vws
   It has the following rights:
   User fhembert:rwx
   Group user:rwx
   Other:: r-x
   ```

2. **Set the view to the newly created view by typing this command:**

   cleartool setview compview

3. Create a VOB to store the elements:

cleartool mkvob -nc -tag /var/tmp/*your-username***_elements_vob **
/var/tmp/*your-username***_elements_vob.vbs**

4 Create a mount point for the VOB that matches the VOB's tag:

mkdir /var/tmp/*your-username***_elements_vob**

5 Mount the VOB:

cleartool mount /var/tmp/*your-username***_elements_vob**.

6 Create a component named **tutcomp** in the elements VOB and associate it with the UCM PVOB:

cleartool mkcomp -nc -root /var/tmp/*your-username***_elements_vob tutcomp@/var/tmp/***your-username***_cctut_ucm_pvob**.

The **-nc** option indicates you do not wish to specify a comment.

The **-root** option specifies what type of component you are creating. As used in the command you entered, it indicates the root directory of the component is **/var/tmp/***your-username***_elements_vob.**

The system's response is similar to this:

```
Set Admin VOB for component "tutcomp"'s VOB
Created component "tutcomp"
```

Using ClearCase LT on UNIX

To create a VOB and its one component in ClearCase, you must perform the following tasks:

1 Create a view using the **cleartool mkview -stgloc** command.

2 Change to the view using the **cd** command.

3 Create a VOB using the **cleartool mkvob** command.

4 Create a component using the **cleartool mkcomp** command.

The exercise guides you in accomplishing these tasks.

Try It!

1 Create a view named compview by typing this command:

cleartool mkview -stgloc compview /var/tmp/compview.vws

```
Created view.
Host-local path: stellar:/var/tmp/compview.vws
Global path: /net/stellar/var/tmp/compview.vws
It has the following rights:
User fhembert:rwx
Group user:rwx
Other:: r-x
```

2 Change to the view:

cd /var/tmp/compview

3 Create a VOB to store the elements:

cleartool mkvob -nc -tag /var/tmp/*your-username***_elements_vob -stgloc \ /var/tmp/***your-username***_elements_vob.vbs**

4 Create a component in the elements VOB and associate it with the UCM PVOB:

cleartool mkcomp -nc -root /var/tmp/*your-username***_elements_vob tutcomp@/var/tmp/***your-username***_tutcomp_ucm_pvob**.

The system's response is similar to this:

```
Set Admin VOB for component "tutcomp"'s VOB
Created component "tutcomp"
```

Where Am I?

So far, you have completed the following tasks:

- Created a PVOB
- Created a component to store the project baseline.
- Created a component for storing the project's elements.

Your next task is that of creating the UCM project.

Creating the UCM Project

You use the New Project wizard, accessible from the Project Explorer, to create a UCM project, and define the policies you want to enforce.

The following sections discuss this task in more detail.

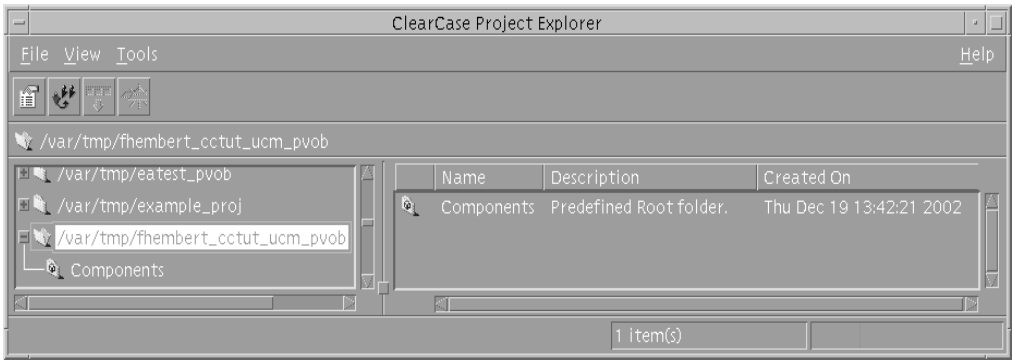
Starting the Project Explorer

1 Start the Project Explorer from the ClearCase Explorer as follows:

- **Windows**. In the left pane of the ClearCase Explorer, click the **Toolbox** tab, then **UCM,** and then click **Project Explorer**.
- **UNIX**. At a system prompt, type **clearprojexp** to start the Project Explorer.

2 The left pane of the Project Explorer lists root folders for all PVOBs in the local ClearCase domain. Each PVOB has its own root folder. ClearCase creates the root folder using the name of the PVOB.

ClearCase also creates a folder called **Components**. This folder contains entries for each component in the PVOB. Folders can contain projects and other folders. Select the root folder for the PVOB that you want to use for storing project information.



3 Click **File > New > Folder** to create a project folder. While it is not necessary to create a project folder, it is a good idea. As the number of projects grows, project folders are helpful in organizing related projects.

Try It!

The Project Explorer may still be running. If not, start it now.

Follow the directions of the platform of interest to you:

- Creating a UCM Project on Windows
- Creating a UCM Project on UNIX

Creating a UCM Project on Windows

1 In the left pane, select the project folder: *your-usename*_**cctut_ucm_pvob**.

2 Click **File > New > Project** to start the New Project Wizard.

The wizard walks you through 5 steps to set up your new project.

Completing Step 1: Naming the Project

In Step 1 of the New Project Wizard, you enter a descriptive name for the project in the **Name** box. You also add a comment in the **Description** box to describe the purpose of this project, and select whether the project you are creating is a single-stream or multiple-stream project.

A traditional parallel development project lets users create multiple streams so that developers can have private and shared work areas. A single-stream project contains only one stream, the integration stream, and users cannot create development streams in single-stream projects.

Try It!

1. Complete Step 1 as follows:
   - **Project name**. Type: **denali_release**
   - **Comment**. Type: **ucm tutorial release**
   - **Stream**. Traditional parallel development.
2. Click **Next**.

Completing Step 2: Deciding the Basis of the Project

Step 2 asks whether you want to create the project based on baselines of an existing project. A baseline identifies one version of every element in a component.

When starting a project from scratch, click **No**.

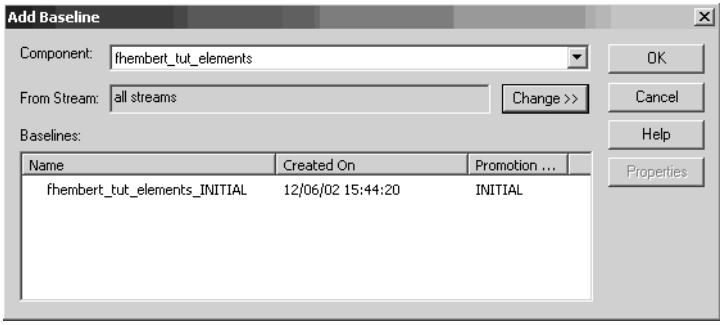
When using an existing project, you select the **Seed this project** option and select the element versions of an existing project that will serve as the basis for the new project.

Try It!

1. You are starting a new project from scratch, so make sure the option **No is selected**.
2. Click **Next**.

Completing Step 3: Choosing the Foundation Baseline

Step 3 asks you to choose the foundation baselines that the project will use. Click **Add** to open the Add Baseline dialog box.



In the **Component** list, select one of the components. The component's initial baseline appears in the **Baselines** list.

Select the baseline and click **OK**.

Try It!

You are going to define the baseline for the *your-username*_**tut_elements_vob** and **tutorial** components.

Complete Step 3 as follows:

1. Click **Add** to open the **Add Baseline** dialog box.

   It may take some time for ClearCase to identify the components. When it is finished, the first component listed is *your-username*_**tut_elements**.

2. In the **Component** list, select the *your-username*_**tut_elements** component.

   The component's initial baseline appears in the **Baselines** list.

3. Click **Change > All Streams**. The baseline box lists **ucm_tut_elements_INITIAL.**

4. Select the component's baseline **ucm_tut_elements_INITIAL**.

5. Click **OK**.

   You now need to repeat the previous steps for the **tutorial** component.

6. Click **Add** again to open the Add Baseline dialog box again.

7. In the **Component** list, select the **tutorial** component. The component's initial baseline appears in the **Baselines** list.

8. Click **Change > All Streams**. The baseline box lists **tutorial_INITIAL.**

9. Select the component's baseline **tutorial_INITIAL**.

10. Click **OK**.

11 Click **Next**.

Completing Step 4: Specify Deliver and View Policies

Step 4 prompts you for the following:

- Make the components modifiable as, by default, they are read-only.
- Define where to deliver the work from the integration stream. You need to set this only if you plan to deliver work to another project.
- Specify deliver and view policies. Policies are rules to enforce development practices among the project team members. By setting policies, you minimize problems you may encounter when integrating the work that developers submit from their private development areas to the project's main shared area.

  For example, you can set a policy that requires developers to update their work areas with the project's latest recommended baseline before they deliver their work. This practice reduces the likelihood that developers will need to work through complex merge procedures when they deliver their work.

  Be aware that, in addition to these, you can create your own policies by using *triggers* on UCM operations. A trigger is a monitor that causes one or more procedures or actions to be executed when a certain ClearCase operation is performed.

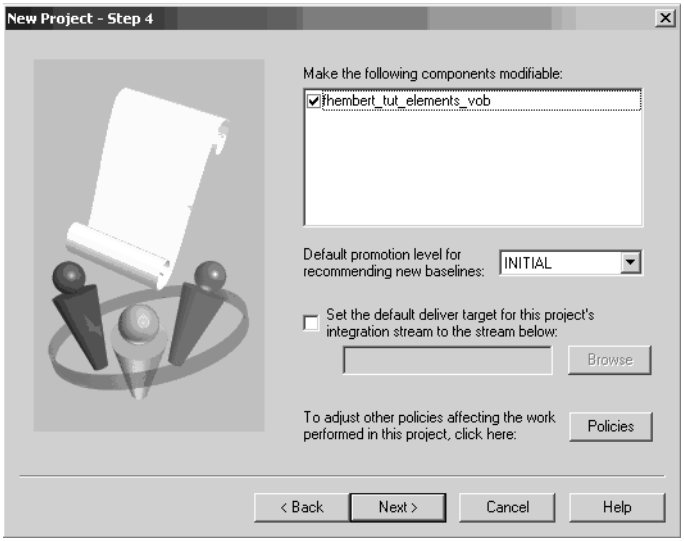
Try It!

Complete Step 4 as follows:

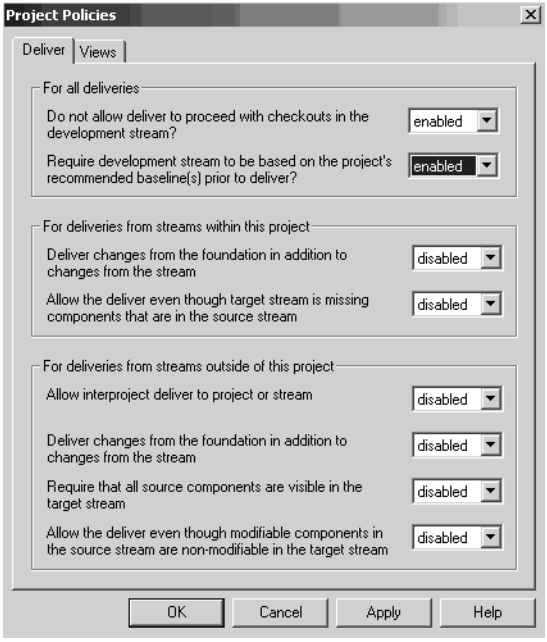
1 In Make the following components modifiable, select the *your-username*_**tut_elements** component.

2 Accept the **INITIAL** promotion level as it makes sense for the start of a new project, even if it is based on the existence of a previous project.

  Click the down arrow to reveal other readily available promotion levels: **REJECTED**, **TESTED**, **BUILT**, and **RELEASED**. ClearCase lets you add your own, and remove any of the default ones. Make sure that **INITIAL** is selected before continuing the exercise.



3 Click the **Policies** button to open the **Project Policies** dialog box where you can specify the deliver policies on the **Deliver** tab and the view policies on the **View** tab for the development team.

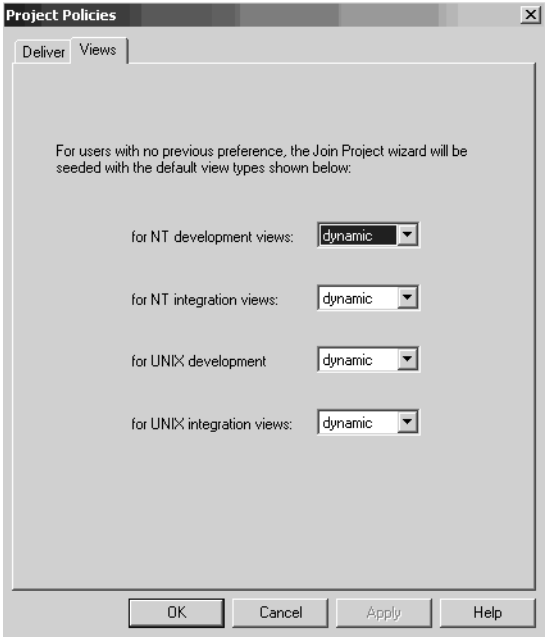


The deliver tab. By default, all deliver policies are disabled.

For this exercise, enable the following policies:

- **Do not allow deliveries to proceed with checkouts in the development stream**. By enabling this policy, you require your developers to check in all files and other elements before starting a deliver operation.
- **Require development stream to be based on the project's recommended baseline prior to deliver**. By enabling this policy, you require that your developers update their private work area with the latest recommended baseline and test their work before delivering their work to the integration stream.

The View tab. On this tab you specify which type of view, dynamic or snapshot, will be created by default the first time your developers join the UCM project using the Join Project Wizard.



When using ClearCase, keep dynamic views for all options.

The Access tab. On this tab you can specify who is allowed to modify the project and its streams.

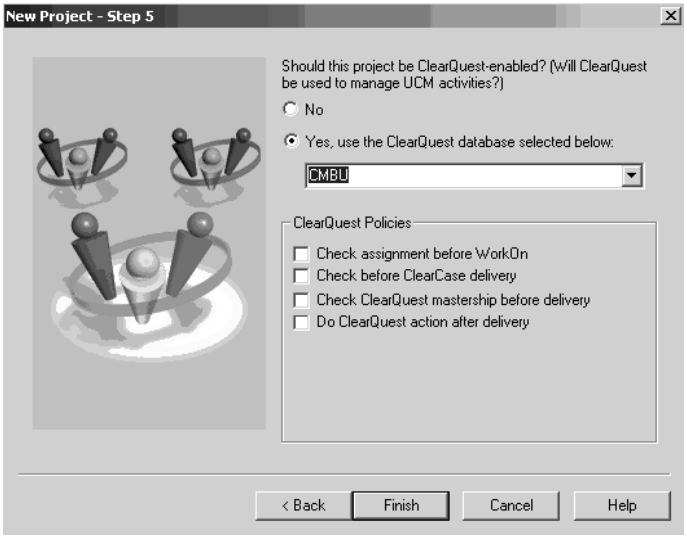
4 Click **OK** to close the Project Policies dialog box.

5 Click **Next.**

Completing Step 5: Integrating with ClearQuest

Step 5 asks whether to configure the project to work with ClearQuest integration. To enable the project to work with Rational ClearQuest, click Yes and select a ClearQuest user database from the list.

When you need to integrate the two products, the list of available ClearQuest policies for each ClearQuest database are listed automatically.

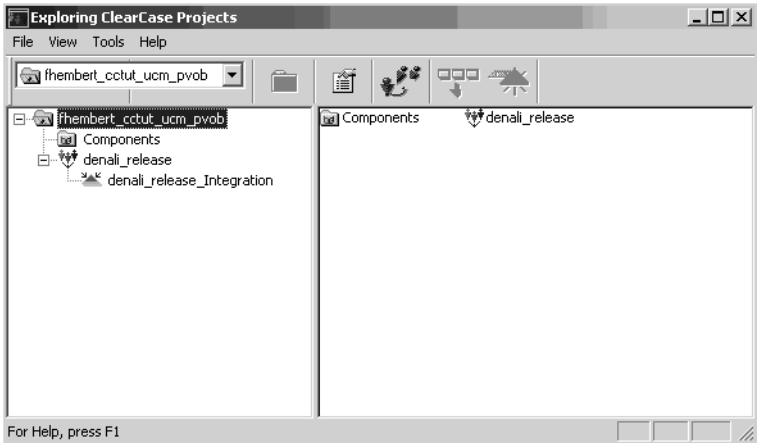


Try It!

To keep the exercise manageable, if ClearQuest is installed on your system, you will not integrate ClearCase with ClearQuest.

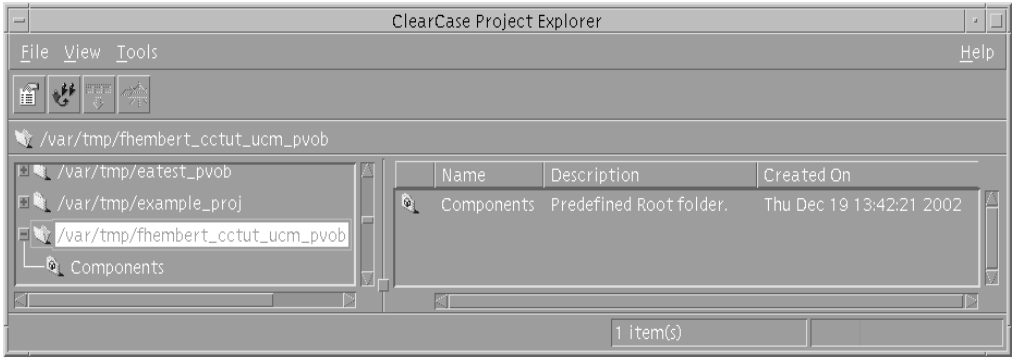
1. Complete Step 5 of the wizard by making sure **No** is selected.
2. Click **Finish**.
3. Review your selections in the confirmation dialog box, and then click **OK**.

   ClearCase creates the UCM project component (**denali_release**) and an integration stream (**denali_release_Integration**).



Creating a UCM Project on UNIX

1. In the left pane, select the UCM project folder: **/var/temp/***your-usename***_cctut_ucm_pvob**.
2. Click **File > New > Project** to start the New Project Wizard.



The wizard guides you through several pages to set up your new project.

Completing Project Information Page

In the **Project Information** page you enter a descriptive name for the project in the **Name** box. You also add a comment in the **Description** box to describe the purpose of this project, and select whether the project you are creating is a single-stream or multiple-stream project.

A traditional parallel development project lets users create multiple streams so that developers can have private and shared work areas. A single-stream project contains

only one stream, the integration stream, and users cannot create development streams in single-stream projects.

Try It!

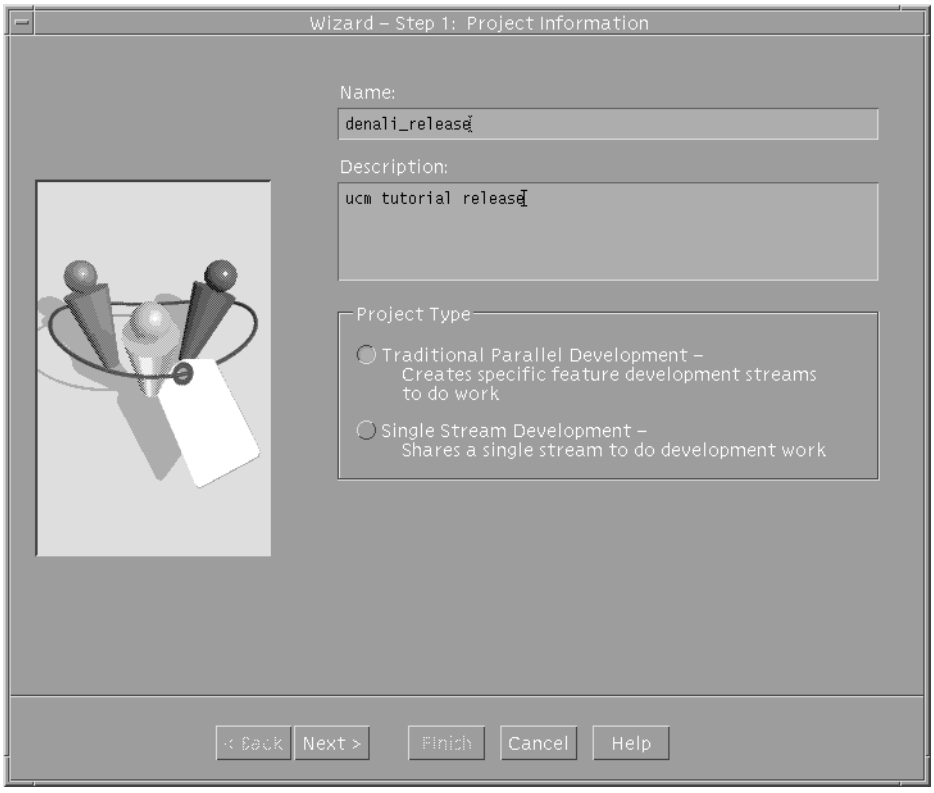
1. Complete the **Project Information** page as follows:
   - **Name**. Type: **denali_release**
   - **Description**. Type: **ucm tutorial release**
   - **Project Type**. Traditional parallel development.
2. Click **Next**.

Completing the Existing Project Page

In the **Existing Project** page you indicate whether you want to create the project based on baselines of an existing project. A baseline identifies one version of every element in a component.

When starting a project from scratch, click **No**.

When using an existing project, you select the **Seed this project** option and select the element versions of an existing project that will serve as the basis for the new project.



Try It!

1 You are starting a new project from scratch, so make sure the option **No is selected**.

2 Click **Next**.

Completing the Component Baselines Page

In the **Component Baselines** page you to choose the foundation baselines that the project will use.

Try It!

You are going to define the baseline for the *your-username*_**tut_elements_vob** and **baslinecomp** components.

Complete **Component Baselines** page as follows:

1 Click **Add** to open the **Add Baselines** dialog box.

It may take some time for ClearCase to identify the components. When it is finished, the first component listed is **baselinecomp**.

2 In the **Component** list, select the **baselinecomp** component.

3 Click the arrow to the right of the streams list, and click All Streams. The baseline box lists **baselinecomp_INITIAL.**

4 Select the component's baseline **baselinecomp_INITIAL**.

5 Click **OK**.

You now need to repeat the previous steps for the **tutcomp** component.

6 Click **Add** again to open the Add Baselines dialog box again.

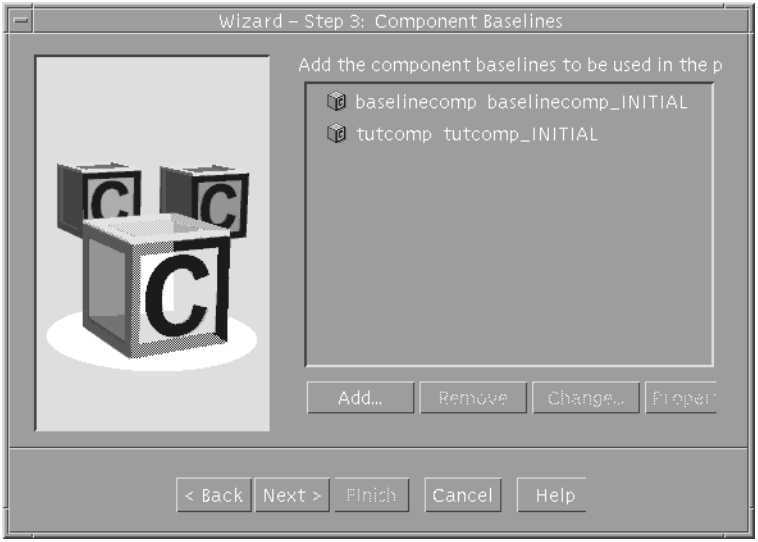
7 In the Component list, select the **tutcomp** component. The component's initial baseline appears in the **Baselines** list.

8 Click the arrow to the right of the streams list, and click **All Streams**. The baseline box lists **tutcomp_INITIAL.**

9 Select the component's baseline **tutcomp_INITIAL**.

10 Click **OK**.

11 Click **Next**.



Completing the Project Policies Setting Page

In the **Project Policies Setting** page you indicate the following:

- Make the components modifiable as, by default, they are read-only.
- Define where work from the integration stream is to be delivered. You need to set this only if you plan to deliver work to another project.
- Specify deliver and view policies. Policies are rules to enforce development practices among the project team members. By setting policies, you minimize problems you may encounter when integrating the work that developers submit from their private development areas to the project's main shared area.

  For example, you can set a policy that requires developers to update their work areas with the project's latest recommended baseline before they deliver their work. This practice reduces the likelihood that developers will need to work through complex merge procedures when they deliver their work.

  Be aware that, in addition to these, you can create your own policies by using *triggers* on UCM operations. A trigger is a monitor that causes one or more procedures or actions to be executed when a certain ClearCase operation is performed.

Try It!

Complete the **Project Policies Setting** page as follows:

1. In Make the following components modifiable, select the **tutcomp** component.
2. Accept the **INITIAL** promotion level as it makes sense for the start of a new project, even if it is based on the existence of a previous project.

   Click the down arrow to reveal other readily available promotion levels: **REJECTED**, **TESTED**, **BUILT**, and **RELEASED**. ClearCase lets you add your own, and remove any of the default ones. Make sure that **INITIAL** is selected before continuing the exercise.
3. Click the **Policies** button to open the **Project Policies** dialog box where you can specify the deliver policies on the **Deliver** tab and the view policies on the **View** tab for the development team.

   The deliver tab. By default, all deliver policies are disabled.

   For this exercise, enable the following policies:

   - **Do not allow deliveries to proceed with checkouts in the development stream**. By enabling this policy, you require your developers to check in all files and other elements before starting a deliver operation.
   - **Require development stream to be based on the project's recommended baseline prior to deliver**. By enabling this policy, you require that your developers update

their private work area with the latest recommended baseline and test their work before delivering their work to the integration stream.

The View tab. On this tab you specify which type of view, dynamic or snapshot, will be created by default the first time your developers join the UCM project using the Join Project Wizard.

- When using ClearCase, select dynamic views for all options.
- When using ClearCase LT, select snapshot views for all options.

The Access tab. On this tab you can specify who is allowed to modify the project and its streams.

4 Click **OK** to close the Project Policies dialog box.

5 Click **Next.**

Completing the ClearQuest Integration Page

In the ClearQuest Integration page you specify if the project is to be integrated with ClearQuest. To enable the project to work with ClearQuest, click Yes and select a ClearQuest user database from the list.

When you need to integrate the two products, the list of available ClearQuest policies for each ClearQuest database are listed automatically.

Try It!

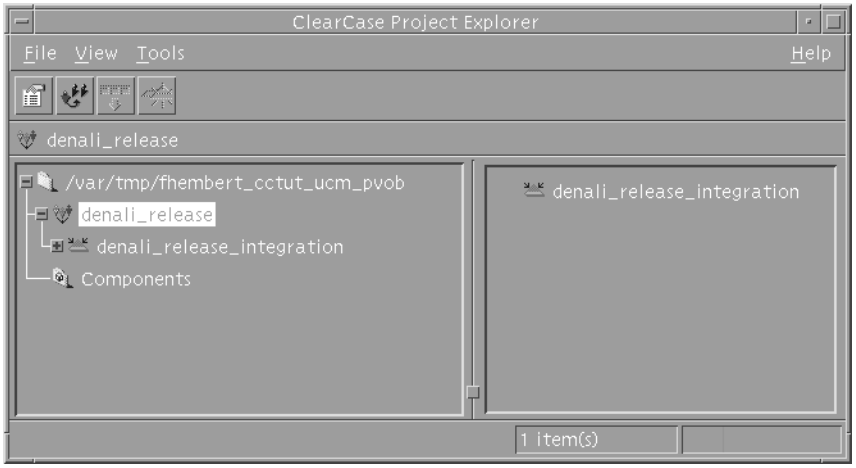
To keep the exercise manageable, if ClearQuest is installed on your system, you will not integrate ClearCase with ClearQuest.

1 Complete the ClearQuest Integration page of the wizard by making sure **No** is selected.

2 Click **Finish**.

3 Review your selections in the confirmation dialog box, and then click **Continue**.

ClearCase creates the UCM project component (**denali_release**) and an integration stream (**denali_release_Integration**).



Where Am I?

So far, you have completed the following tasks:

- Created a PVOB
- Created a component to store the project baseline.
- Created a component for storing the project's elements.
- Created the UCM project.

Your next task is to create a view to see the elements in the integration stream.

Creating an Integration View

When you create a UCM project, ClearCase creates the project's integration stream for you. An integration stream is a ClearCase UCM object that enables access to versions of the project's shared elements.

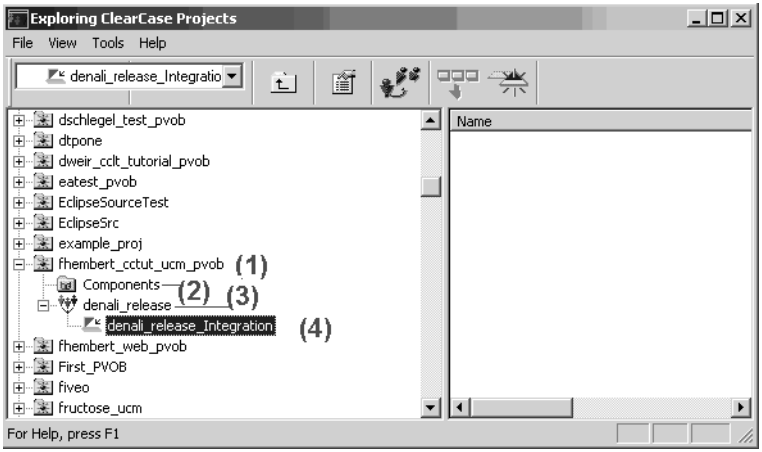
By default, a project contains only one integration stream, which maintains the project's baselines. The integration stream configures integration views to select the versions associated with the foundation baselines plus any activities and versions that have been delivered.

To see and make changes to the project's shared elements, you need an *integration view.* To create an integration view, follow these steps.

1 In the Project Explorer, navigate to the integration stream by moving down the object's hierarchy:

   - Root folder (1)

- Project folder (2)
- Project (3)
- Stream (4)



2 Select the integration stream and click **File > New > View** to start the View Creation Wizard.

3 In the View Creation Wizard, accept the default values to create an integration view attached to the integration stream. By default, the View Creation Wizard uses this convention for the integration view name: *username_project-name_***integration**.

ClearCase supports dynamic and snapshot views. ClearCase LT supports only snapshot views.

Dynamic views use the ClearCase multiversion file system (MVFS) to provide immediate, transparent access to files and directories stored in VOBs. ClearCase maps a dynamic view to a drive letter in Windows Explorer. We recommend that you make the integration view a dynamic view to ensure that you always see the correct version of files and directories that developers deliver to the integration stream.

Snapshot views copy files and directories from the VOB to a directory on your computer. When using snapshot views, you must perform an *update* operation to copy the latest delivered files and directories to your computer.

Follow the directions for the platform of interest to you:

- Creating an Integration View on Windows
- Creating an Integration View on UNIX

Creating an Integration View on Windows

Try It!

1 In the Project Explorer, navigate to the integration stream:

your-username_cctut_ucm_pvob > Components > denali_release > denali_release_Integration

2 Select the **denali_release_Integration** stream, right-click and select **Create View** to start the View Creation Wizard.

The View Creation wizard guides you through two pages of options.

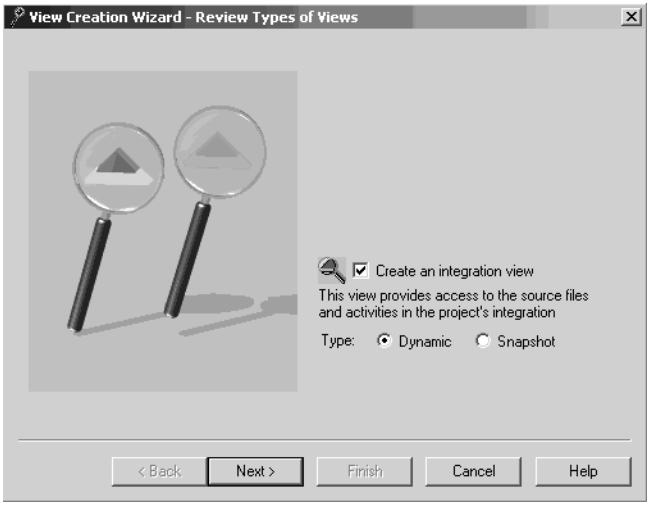
3 Complete the **Review Types of Views** page of the wizard as follows:

a Verify the option **Create an integration view** is selected.

b Verify the option **Dynamic** is selected by default.

ClearCase users: keep these defaults.

ClearCase LT users: select **Snapshot** as type of view. Keep the default value for the integration view.



Click **Next**.

4 Complete the Choose Name and Drive page of the wizard as follows:

a Accept the default view name: *your-username*_**denali_release_int**.

b Accept the default drive letter. For example: **Z**

c Add a comment: **Integration view for denali_release**.

d Click **Advanced Options** and verify that the storage location of the integration view is similar to the following:

Z:*your-computer-name*\cc-tut*your-username*_denali_release_integration.vws.

If not, click **Browse** and navigate to the integration view drive.

e Verify that the options **Use interop text mode** and **Create shareable derived objects** are selected.

f Click **OK** to close the dialog box.

g Click **Finish**.

h On the Confirm dialog box, click **Details** to view the selections you have made, and then click **Close** to close the View Details dialog box.

i The View Creation Status dialog box reports that the creation of the integration view was successful. Click **Close** to close the dialog box.

5 ClearCase creates a **View** tab in the ClearCase Explorer. If the ClearCase Explorer is up and running, you must update your view shortcuts as follows:

a In the ClearCase Explorer, and press ALT+F5 (or **View > Refresh View Shortcuts**).

b ClearCase found the new view and created a **denali_release** shortcut button.

c Click the **denali_release** shortcut button to reveal the integration view to the denali_release integration stream: *your-username*_**denali_release_integration**.

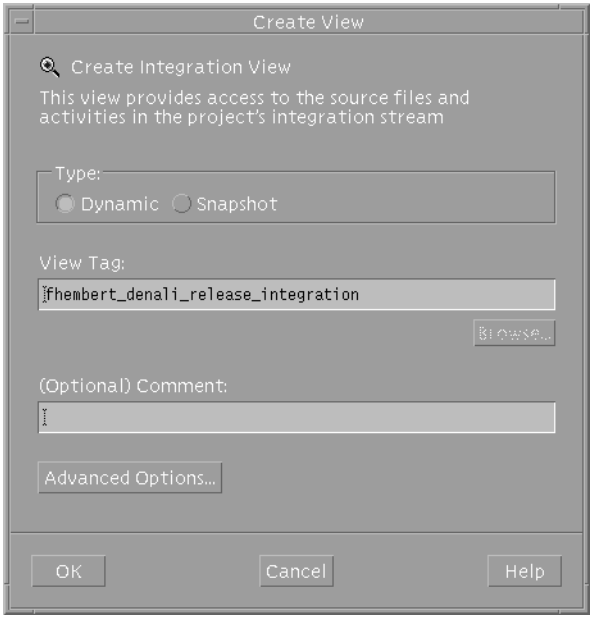
Creating an Integration View on UNIX

Try It!

1 In the Project Explorer, navigate to the integration stream:

*your-username*_**cctut_ucm_pvob > Components > denali_release > denali_release_Integration**

2 Select the **denali_release_Integration** stream, right-click and select **Create View** to open the Create View dialog box.



3 In the Create View dialog box, do the following:

a Verify the type of view selected is **Dynamic.**

ClearCase LT users: **Snapshot** is selected as the type of view.

b Verify the view tag listed is *your-username*_**denali_release_intergration**.

c Click **Advanced Options** and verify that the storage location of the integration view is similar to the following:

/net/*your-system***/export/home/***your-username***/***your-username*_**denali_release_integration.vws**.

d Verify that the options **Use interop text mode** and **Create shareable derived objects** are selected.

e Click **OK** to close the dialog box.

f On the Confirm dialog box, click **Continue**.

The View Creation Status dialog box reports that the creation of the integration view was successful.

g Click **Close**.

Where Am I?

So far, you have completed the following tasks:

- Created a PVOB
- Created a component to store the project baseline.
- Created a component for storing the project's elements.
- Created the UCM project.
- Created a view to see the elements in the integration stream.

UNIX users. Your next task is to create and set a UCM activity.

Windows users. Your next task is to create a directory structure and add it to source control.

Creating and Setting UCM Activities (UNIX Only)

Before you can create a directory structure and add the elements under ClearCase source control, you need to to create and set a UCM activity.

To accomplish this, you perform the following tasks:

1 Set your integration view if it is a dynamic view using the **cleartool setview** command.

2 Create an activity using the **cleartool mkactivity** command.

The following sections guide you in creating an UCM activity.

Try It!

1 The first task is to set your integration view. At the system prompt of a terminal window, type:

cleartool setview *your-username*_**denali_release_integration**

2 The activity you are going to create is one for creating a directory. To create the activity **create_directories**, type the following command:

cleartool mkactivity -headline "Create directories" create_directories

The system's response is similar to the following:

```
Created activity "create_directories"

Set activity "create_directories" in view
"fhembert_denali_release_integration".
```

Creating a Directory Structure and Adding to Source Control

When creating the project from scratch, you need to create the directory elements within the project's components to implement the directory structure that you defined for your project.

Follow the instructions for the platform of interest to you:

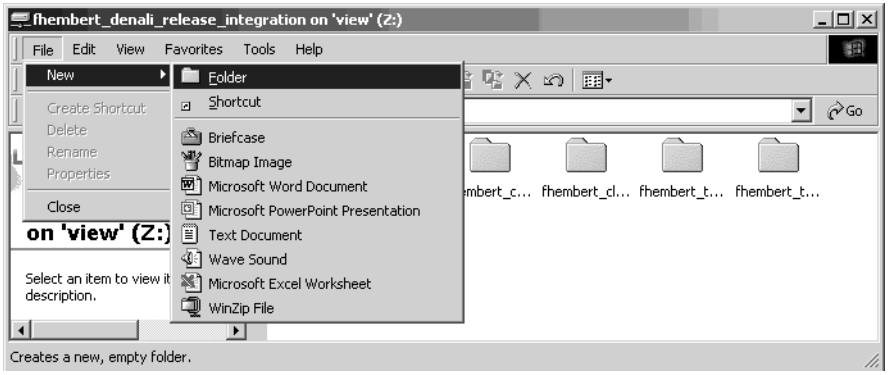
- Creating a Directory Structure on Windows
- Creating a Directory Structure on UNIX

Creating a Directory Structure on Windows

On Windows, new directories are created using the Windows Explorer.

Try It!

You are now going to create a folder named Project Sources in the component *your-username*_**tut_elements_vob** on the drive where the integration view is located.



1 Double-click the **My Computer** icon on your desktop.

2 Double-click the integration view; it is similar to the following: *your-username*_**denali_release_int on 'View' (Z:)**.

For ClearCase LT users, the *your-username*_**tut_elements_vob** is located on your **C:** drive. You will create a new folder under this directory.

3 Double-click the *your-username*_**tut_elements_vob** folder to open it.

4 Click **File > New > Folder** and type the name **Project_Sources**.

5 To add the new folder under ClearCase source control, right-click **Project_Sources** and select **Add to Source Control** from the shortcut menu.

6 Complete the **Add to Source Control** dialog box as follows:

   a As the activity to be associated with the creation of the new folder, type: **set up folder to hold denali release sources**.

   b Keep the option **Checkout after adding to source control cleared. In the context of the exercise, you do not want ClearCase to check out the folder after adding it to source control.**

   c Select the option **Preserve file modification time** to create a time record for each time the folder is being checked out and then checked in again.

   d Click **OK** to close the dialog box.

   The **Project_Sources** folder is being created and put under source control.

If using a snapshot view, you also need to update your integration view:

1 Go to the ClearCase Explorer.

2 Select the *username*_**denali_release_integration** view and press ALT+F5 to update the view.

3 In the folders pane, double-click the **ucm_tutorial** folder where you will see the folder **Project_Sources** is under source control.

4 Close the **Windows Explorer**.

Creating a Directory Structure on UNIX

To create a directory structure on UNIX, your integration view must be set to an activity. In the section *Creating and Setting UCM Activities (UNIX Only)* you create an activity "create_directories" and this activity is used when creating the directory structure for the **denali_release** project.

Try It!

1 Navigate to the root directory of the *your-username*_**elements_vob** VOB, which is **/var/tmp/,** by typing this command:

cd /var/tmp/

2 Check out the *your-username*_**elements_vob directory using the cleartool checkout command:**

cleartool co -co "Creating directory structure" *your-username*_**elements_vob**

co is the short form of the checkout command.

-co is the short form of the **-comment** option.

The system's response is similar to the following:

```
Created branch "denali_release_integration" from
"fhembert_element_vob" version "/main/denali_release_integration/0"

Checked Out "fhembert_element_vob" from version
"/main/denali_release_integration/0"

Attached activities:

activity:create_directories@/var/tmp/fhembert_cctut_ucm_pvob
"Create Directories"
```

3 Go to the *your-username*_**elements_vob** directory using the **cd** command:

cd *your-username*_**elements_vob**

4 You are now going to create a directory named design using the **cleartool mkelem** command. Type the following:

cleartool mkelem -co "Prototype directory" -eltype directory design

The mkelem command creates an element.

The **-eltype** option, meaning **element type**, specifies that the element to be created is a directory.

design is the name of the directory to be created.

The system's response is similar to the following:

```
Created element "design" (type "directory")

Created branch "denali_release_integration" from "design" version
"/main/0".

Checked out "design" from version
"/main/denali_release_integration/0".

Attached activities:

activity:create_directories@/var/tmp/fhembert_cctut_ucm_pvob
"Create Directories"
```

ClearCase leaves the directories checked out, so you need to check them in.

5 Check in the design directory using the **cleartool checkin** command:

cleartool checkin -nc design

6 Navigate to the *your-username*_**element_vob** directory by moving up one level:

cd ..

7 Now check in the root directory:

cleartool checkin -nc *your-username*_**element_vob**

Where Am I?

So far, you have completed the following tasks:

- Created a PVOB.
- Created a component to store the project baseline.
- Created a component for storing the project's elements.
- Created the UCM project.
- Created a view to see the elements in the integration stream.
- UNIX users created and set a UCM activity.
- Created a directory structure and added it to ClearCase source control.

Your next, and final, task is to define an integration baseline so that UCM developers can join the UCM project.

Defining an Integration Baseline (Windows and UNIX)

To complete the set-up of a UCM project environment, you need to define an integration to allow the developers to join the UCM project you created and start working.

You define the integration baseline using the Project Explorer.

This final task also sets the foundation for completing the exercises of the module *Joining a UCM Project*.

Try It!

1 If you need to start the Project Explorer again, do so by clicking **Start > Programs > Rational Software > Rational ClearCase > Project Explorer**.

2 In the Project Explorer, select the integration stream **denali_release_integration**.

3 Right-click and select **Make Baseline** from the shortcut menu to open the Make Baseline dialog box.

4 Make the following selections in the Make Baseline dialog box:

   a Accept the default base name (in UNIX: Baseline title); it should be similar to this: **denali_release_12_10_2002**.

   b Add an optional comment: initial integration baseline.

   c Make sure **incremental** is selected as the type of baseline. An incremental baseline is a baseline that ClearCase creates by recording the last full baseline and those versions that have changed since the last full baseline was created.

   A full baseline is a baseline that ClearCase creates by recording all versions below the component's root directory.

   d Verify that the project stream lists: **denali_release/denali_release_Integration**.

   e Verify that the view context lists the following:

   On Windows: *your-username*_**denali_release_int**

   On UNIX: *your-username*_**denali_release_integration**

   f Click **OK**.

   g ClearCase reports that one new baseline was created. Click **OK (on UNIX: Dismiss)** to close the message box.

   h Click **Close**. (Windows only.)

Where Am I?

You have completed all the tasks required for setting up a UCM project environment. UCM developers can now join the UCM project.

Joining a UCM Project

11

Audience	Developers
Platforms	Windows, UNIX
Process	UCM
Products	ClearCase, ClearCase LT

Note: The exercises in this module rely on the results of the exercises in the module *Setting Up a UCM Project*.

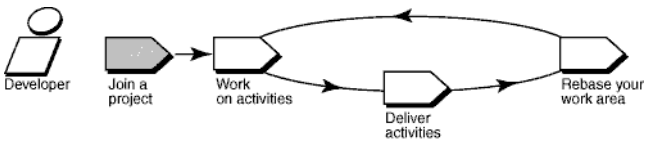
When a development team adopts the Unified Change Management (UCM) process for managing projects, project managers first set up the project's environment by creating the data repositories, determining the location of resources, and the project infrastructure and policies that will govern the development effort.

After project managers have established the UCM project environment, developers can then set up their work areas. This is accomplished by joining the UCM project.

This module covers the following topics:

- About the UCM Developer Tasks
- Joining a UCM Project

About the UCM Developer Tasks



When you join a UCM project, your work as a developer involves the following tasks:

1 Join the UCM project to set up your work area.

2 Work on your activities in your private work area. You access the sources in your private work area through your development view.

3 Deliver your work to the shared work area. You access the sources in the shared work area through your integration view.

4 Rebase your work area.

To access your project's source files, you set up your work area by joining a UCM project. As you modify source files, you use activities to organize and identify your work. Other developers on the project do not see your work until you deliver it to a shared work area.

Periodically, the project manager incorporates activities in the shared work area into baselines. A baseline selects one version of every element in a component.

When project managers announce that a new recommended baseline is available, you synchronize (or rebase) your work area with the activities in the new baseline.

This module focuses on the first developer's task: Joining a UCM project.

Joining a UCM Project

Joining a UCM project involves the following tasks:

- Starting the Join Project Wizard
- Choosing the Project
- Creating a Development Stream
- Creating Views for Your Development and Integration Streams
- Specifying the Location for Your Development View
- Specifying the Location for Your Integration View
- Selecting the Project's Components

The following sections take you through the task of joining a UCM project.

Follow the instructions for the platform of interest to you:

- Joining a UCM Project on Windows
- Joining a UCM Project on UNIX

Joining a UCM Project on Windows

Starting the Join Project Wizard

1 Start the ClearCase Explorer by clicking the ClearCase Explorer icon on your desktop.

2 Click the **Toolbox** tab and then click the **UCM** tab.

3 Click **Join Project** to start the Join Project Wizard.

The wizard guides you through six pages of options that create your private work environment and views so that you can access the sources and start your work.

Try It!

1 If necessary, start the ClearCase Explorer by clicking the ClearCase Explorer icon on your desktop.

2 Click the **Toolbox** tab and then click the **UCM** tab.

3 Click **Join Project** to start the Join Project Wizard.

4 Continue by following the directions for the product you are using:

- Creating a UCM Project Using ClearCase
- Creating a UCM Project Using ClearCase LT

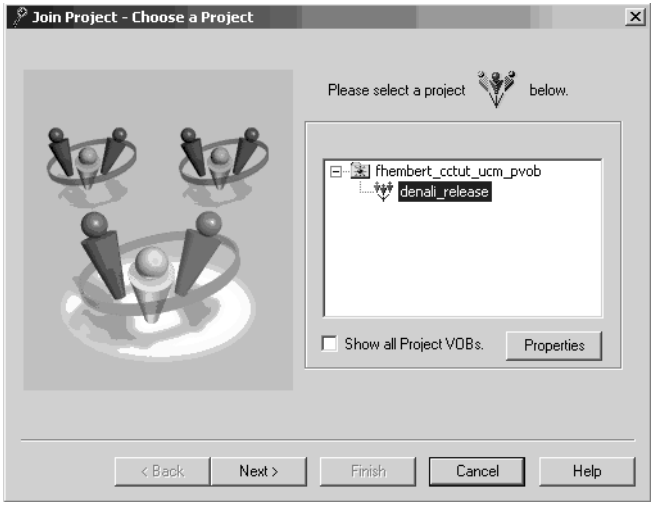
Creating a UCM Project Using ClearCase

Choosing the Project

The first page of the Join Project Wizard lists the projects that can be joined.

From that list, select the project you want to join.

As several projects can be grouped under one project folder, you may need to expand the project folder to find the project you want to join.



Try It!

The first page of the wizard lists the UCM project: **denali_release** project. This project was created during the exercises in the module *Setting Up a UCM Project*.

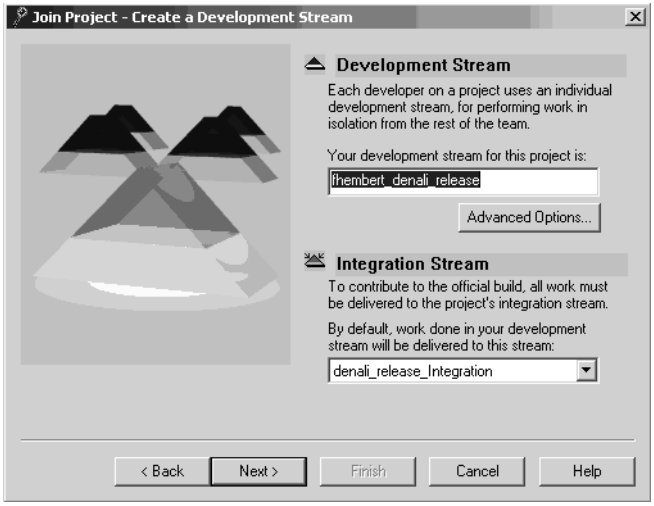
1 Select the UCM project **denali_release.**

2 Click **Next**.

Creating a Development Stream

In the second page of the Join Project Wizard you will select a name for the development stream and identify the stream to which you will deliver your work. By default, this is the integration stream.

During the project environment setup, ClearCase created a projectwide, shared integration stream. When you join a project, ClearCase creates a development stream to track your work.



1 In Development Stream, specify a name for your development stream.

A useful naming convention is to include your username in the development stream's name. For example: *jdoe*_**denali_release**, where *jdoe* is a developer's username and **denali_release** is the name of the project you are joining.

2 Identify the integration stream of the project you are joining and select it from the list.

Try It!

On the second page of the wizard, ClearCase automatically completed the developer's stream field by providing a default name for your developer's stream.

In ClearCase, notice the name format: *your-username_project-name*. In our exercise, this corresponds to *your-username*_**denali_release**. Accept the developer's stream.

In ClearCase LT, it corresponds to *your-username*_**cctut_ucm_pvob**.

3 ClearCase also automatically completed the integration stream field with the name of the project's integration stream.

In ClearCase: **denali_release_Integration**.

In ClearCase LT: **IntegrationStream**.

4 Accept the integration stream, and click **Next**.

Creating Views for Your Development and Integration Streams

The third page of the Join Project Wizard prompts you to specify the type of view you want to use for the development and integration streams. Views are the mechanism by which you get access to the project's sources.

In this stage of joining a project, two views are created: one for accessing the sources in your development stream, a second one for accessing the sources in the integration stream. For each stream, you need to specify the type of view you want. You can choose to have two views of the same type, for example, two dynamic views or two snapshot views; or you can have a mix of one dynamic view and one snapshot view.

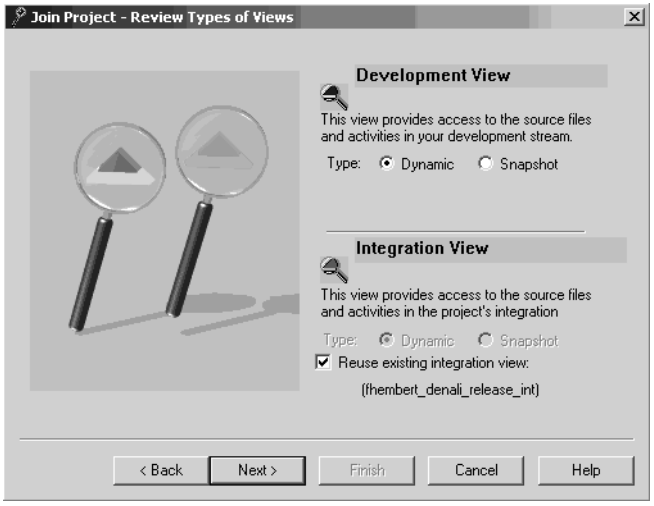
Depending on the product you use, your options are the following:

- For ClearCase: dynamic and snapshot views.
- For ClearCase LT: snapshot views only.

Dynamic views provide access to the most up-to-date version of the project's element in a dynamic fashion.

Snapshot views provide access to the project's elements at a static point in time. To ensure that you are looking at the most recent version, you need to periodically perform an **update** operation to update your environment.

To specify the views you want, select the appropriate options.



Try It!

On the third page of the wizard, specify the following:

1 For the development view:

- ClearCase users: select **dynamic** for both views.
- ClearCase LT users: select **snapshot** for both views.

For the integration view, ClearCase automatically selected the reuse of the integration view that was created when you performed the exercises of the module *Setting Up a UCM Project*.

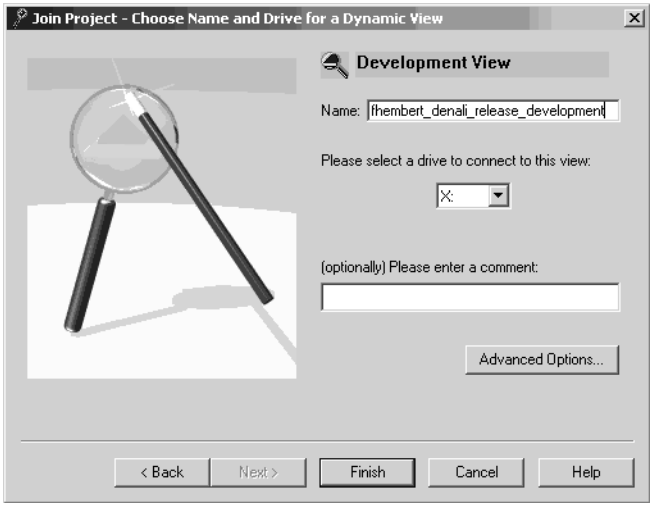
2 Accept the listed integration view: *your-username*_**denali_release_int**.

3 Click **Next**.

Specifying the Location for Your Development View

In the fourth page of the Join Project Wizard, you specify where the view for your development stream will be located.

You can accept the default value, or you can specify a directory location on a remote host or locally on your computer. If the directory does not exist, ClearCase informs you and prompts you to confirm that you agree to creating the directory.



Try It!

1 Complete the fourth page of the wizard by modifying the default view name to read: *your-username*_**denali_project_development**.

When you are new to ClearCase, adding the word *development* is a visual aid to distinguish the development stream view from the integration stream view.

2 Accept the suggested drive. For example: Y:

3 You want the view to be located on your computer instead of on a network host. To specify the location of the view, click **Advanced Options**.

4 Select the option **Use Explicit Path**.

5 Click the text box and modify the folder in which the view is to be located. The path should be similar to the following:

*your-username***cc-tut***your-username*_**denali_release_development.vws**.

Specifying the Location for Your Integration View

The fifth page is similar to the fourth page and appears only if you chose to create a new integration view. In this page you specify the view location for the project's integration stream.

In the tutorial, this page does not appear.

Selecting the Project's Components

The sixth and final page of the Join Project Wizard appears only when you work with snapshot views or use ClearCase LT.

In the **Components** page of the wizard, you select the project's components that you want ClearCase to populate in both your development and integration views.

1 Select the components to be loaded in your views from the **Choose Component** list.

2 Click **Finish**.

A Confirm dialog box lists the locations you selected or specified for your development and integration views.

3 Optionally, you can get details about the selections you made for each view by clicking **Details**.

4 On the Confirm dialog box, click **OK**.

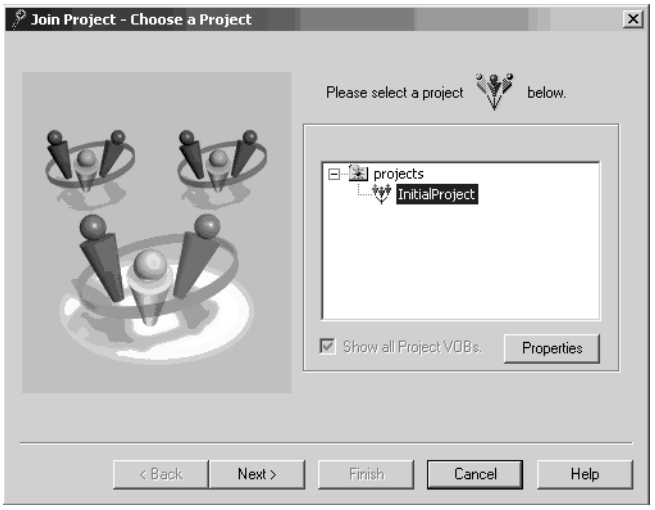
As a member of the project you just joined, your environment has been set up so that you are ready to start your work.

Creating a UCM Project Using ClearCase LT

Choosing the Project

The first page of the Join Project Wizard lists the projects that can be joined.

From that list, select the project you want to join.



As several projects can be grouped under one project folder, you may need to expand the project folder to find the project you want to join.

Try It!

The first page of the wizard lists the UCM project: *your-username*_**cctut_ucm_pvob** project.

1 Select the UCM project *your-username*_**cctut_ucm_pvob.**

2 Click **Next**.

Creating a Development Stream

In the second page of the Join Project Wizard you will select a name for the development stream and identify the stream to which you will deliver your work. By default, this is the integration stream.

During the project environment setup, ClearCase created a projectwide, shared integration stream. When you join a project, ClearCase creates a development stream to track your work.

1 In Development Stream, specify a name for your development stream.

A useful naming convention is to include your username in the development stream's name. For example: *jdoe*_**denali_release**, where *jdoe* is a developer's username and **denali_release** is the name of the project you are joining.

2 Identify the integration stream of the project you are joining and select it from the list.

Try It!

On the second page of the wizard, ClearCase automatically completed the developer's stream field by providing a default name for your developer's stream. It corresponds to *your-username*_**cctut_ucm_pvob**.

1. ClearCase also automatically completed the integration stream field with the name of the project's integration stream: **IntegrationStream**.
2. Accept the integration stream, and click **Next**.

Specifying the Location for Your Snapshot View

In this page of the wizard, you specify the location for your snapshot view.

Try It!

1. Click **Browse**.
2. Navigate to this location:

C:\ClearCaseStorage\Views*your-domain\your-username\your-username*_**cctut_ucm_pvob.vws**

3. Click **OK**.
4. Click Next.

Specifying the Location for Your Development View

In this page of the Join Project Wizard, you specify where the view for your development stream will be located.

Try It!

1. Accept the default view name.
2. Type as comment: Development view.
3. Click **Next**.

Specifying the Location for Your Integration View

Here you specify the location of your integration view.

Try It!

1. Accept the default view name.
2. Type as comment: Integration view.
3. Click **Next**.

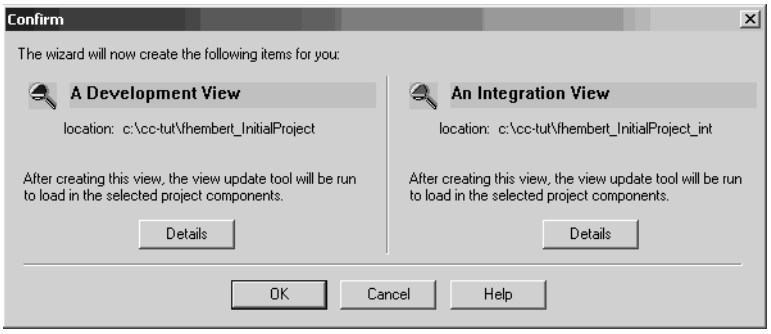
Selecting the Project's Components

In the **Components** page of the wizard, you select the project's components that you want ClearCase to populate in both your development and integration views.

1 Select the components to be loaded in your views from the **Choose Component** list.

2 Click **Finish**.

A Confirm dialog box lists the locations you selected or specified for your development and integration views.



3 Optionally, you can get details about the selections you made for each view by clicking **Details**.

4 On the Confirm dialog box, click **OK**.

As a member of the project you just joined, your environment has been set up so that you are ready to start your work.

Try It!

1 Accept the **cclt_tutorial_sources** component.

2 Click **Finish**.

3 In the Confirmation dialog box, click **OK**.

4 In the View Creation Status dialog box, click **OK**.

Joining a UCM Project on UNIX

Starting the Join Project Wizard (UNIX)

You start the Join Project Wizard by typing **clearjoinproj** at the system prompt in a terminal window.

Try It!

Start the Join Project Wizard.

Selecting the Project

The first page of the Join Project Wizard lists the projects that can be joined.

From that list, select the project you want to join.

As several projects can be grouped under one project folder, you may need to expand the project folder to find the project you want to join.

Try It!

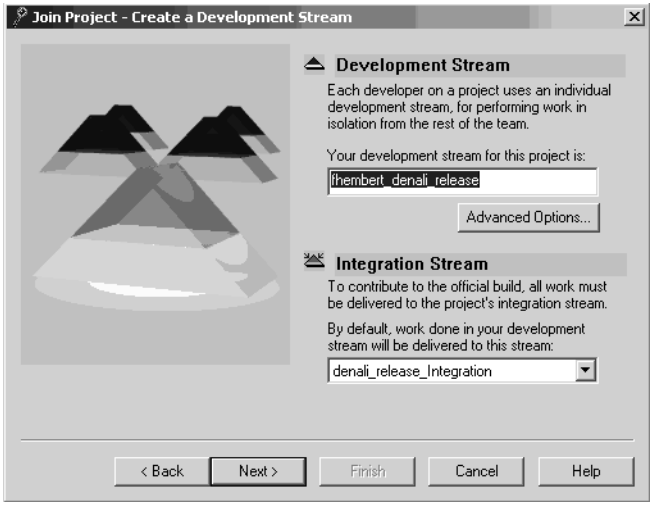
The first page of the wizard lists the **denali_release** project. You may need to expand the *your-username*_**cctut_ucm_pvob** to see the denali_release project. This project was created during the exercises in the module *Setting Up a UCM Project*.

1. Select the **denali_release** project.
2. Click **Next**.

Creating a Development Stream

In the second page of the Join Project Wizard you will select a name for the development stream and identify the stream to which you will deliver your work. By default, this is the integration stream.

During the project environment setup, ClearCase created a projectwide, shared integration stream. When you join a project, ClearCase creates a development stream to track your work.



1 In Development Stream, specify a name for your development stream.

   A useful naming convention is to include your username in the development stream's name. For example: *jdoe*_**denali_release**, where *jdoe* is a developer's username and **denali_release** is the name of the project you are joining.

2 Identify the integration stream of the project you are joining and select it from the list.

Try It!

1 On the second page of the wizard, ClearCase automatically completed the developer's stream field by providing a default name for your developer's stream. Notice the name format: *your-username_project-name*. In our exercise, this corresponds to *your-username*_**denali_release**. Accept the developer's stream.

2 ClearCase also automatically completed the integration stream field with the name of the project's integration stream: **denali_release_Integration**. Accept the integration stream.

3 Click **Next**.

Creating Your Development and Integration Views

In this stage of joining a project, two views are created: one for accessing the sources in your development stream, a second one for accessing the sources in the integration stream. For each stream, you need to specify the type of view you want.

You can choose to have two views of the same type, for example, two dynamic views or two snapshot views; or you can have a mix of one dynamic view and one snapshot view.

Depending on the product you use, your options are the following:

- For ClearCase: dynamic and snapshot views.
- For ClearCase LT: snapshot views only.

Dynamic views provide access to the most up-to-date version of the project's element in a dynamic fashion.

Snapshot views provide access to the project's elements at a static point in time. To ensure that you are looking at the most recent version, you need to periodically perform an **update** operation to update your environment.

Creating Your Development View

The third page of the Join Project Wizard prompts you to specify the type of view you want to use for your development stream. Views are the mechanism by which you get access to the project's sources.

Try It!

1. Depending on the product you use, select the following type of view:
   - For ClearCase: dynamic view.
   - For ClearCase LT: snapshot views only.
2. The view tag listed is *your-username*_**denali_release**. Modify the view tag to read: *your-username*_**denali_release_development**
3. Click **Next**.

Creating your Integration View

The fourth page of the Join Project Wizard prompts you to specify the type of view you want to use for your integration stream.

This page is similar to the page in which you specified the view for your development stream.

Try It!

1. Select the same type of view as for your development view.
   - For ClearCase: dynamic view.
   - For ClearCase LT: snapshot views only.
2. Click **Finish**.

3 In the Confirm dialog box, click **Continue**.

ClearCase provides feedback on the status of creating the two views, and opens the Start Development View dialog box.

4 In the Start Development View dialog box, click **Done**.

You have successfully accomplished the tasks required to join a UCM project.

Mounting and Unmounting VOBs

12

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase only

To access the files in your dynamic and snapshot views, the VOBs in which the files reside must be mounted on your system. A VOB, or Versioned Object Base, is the ClearCase term for a data repository, or database. In ClearCase, all files and directories are stored in VOBs.

This module covers the following topics:

- Mounting VOBs
- Unmounting VOBs

Mounting VOBs

On Windows, you mount VOBs from the following places:

- From the ClearCase Explorer
- From the Windows Explorer
- From the command line

On UNIX, you mount VOBs from the command line.

NOTE: The tutorial explains mounting and unmounting VOBs only from the ClearCase Explorer on Windows, and from the command line on UNIX.

From the ClearCase Explorer (Windows)

1. Select the root directory of the view.
2. Right-click and select **Mount VOB** from the shortcut menu to open the Mount dialog box.

3 Click the VOBs you want to mount. You can mount contiguous and noncontiguous VOBs.

4 By default, the option **Reconnect at Logon** is selected. This ensures that when you log on to your system, the VOBs you select will be mounted automatically.

5 Click **OK**. The VOBs are mounted and become accessible through your views.

From the Command Line (UNIX)

1 At the system prompt, type this command:

- To mount one VOB: **cleartool mount** *vob-tag*

  For Windows platforms, the *vob-tag* is the VOB's registered name and also its root directory—the pathname at which you as a user access VOB data.

  For UNIX platforms, the *vob-tag* is the full pathname at which you as a user access a VOB.

  When using Full ClearCase with dynamic views, the VOB storage directory is activated by mounting it as a file system of type MVFS at the location specified by its VOB-tag.

  On Windows, when using ClearCase LT, a VOB-tag has a single component and begins with the backslash (\). For example, **\myvob** and **\vob_project2** are valid VOB-tags.

- To mount all public VOBs: **cleartool –all**

Unmounting VOBs

The procedures for unmounting VOBs are similar to the procedures for mounting VOBs.

On Windows, you unmount VOBs from the following places:

- From the ClearCase Explorer
- From the Windows Explorer
- From the command line

On UNIX, you unmount VOBs from the command line.

From the ClearCase Explorer (Windows)

1 Select the root directory of the view.

2 Right click and select **Unmount VOB** from the shortcut menu. The Unmount dialog box opens.

3 Click the VOBs you want to unmount.

4 Click **OK**. The VOBs are unmounted and are no longer accessible through your views.

From the Command Line (UNIX)

At the system prompt, type this command:

To unmount one VOB: **cleartool umount** *vob-tag*

To unmount all public VOBs and all private VOBs that you own: **cleartool umount –all**

Try It!

On Windows

1 Go the ClearCase Explorer and click the *your-username*_**denali_release_development view.**

2 *Select the your-username*_**denali_release_development root directory**.

3 Right-click and select **Unmount VOB** from the shortcut menu.

4 In the Unmount dialog box, select the **ucm_tutorial** VOB and click **Unmount**. The VOB is removed from the folders pane.

5 Now, mount the VOB again. If necessary, *select the your-username*_**denali_release_development root directory**.

6 Right-click and select **Mount VOB** from the shortcut menu.

7 From the list, select the **ucm_tutorial** VOB and click **Mount.** If there are many VOBs, you may need to scroll to the bottom of the list to find it. The VOB appears again in the folders pane.

On UNIX

To unmount, type this command:

cleartool umount *your-username*_**denali_release_development**

To mount the VOB again, type:

cleartool mount *your-username*_**denali_release_development**

Working with Views 13

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

A view is the mechanism ClearCase uses to provide access to specific versions of elements under source control. Rules determine which version of each element is visible and accessible through the view. As such, views are indispensable when working with Rational ClearCase and Rational ClearCase LT.

This module covers the following topics:

- About Views and View Directories
- Types of Views
- Naming Your Views
- Which View to Create
- Creating a View
- Starting a View
- Stopping a View
- Updating a Snapshot View
- Removing Views

About Views and View Directories

A view provides access to one version of each element in your project. In ClearCase Explorer (Windows), the view presents the elements in a tree format in the folders pane.

In UCM, a developer's work area contains two views:

- **Development view**. This view represents the view to the development stream or branch, which gives access to the elements a developer is working on. In this view,

the developer checks out and modifies source elements, compiles them into object modules for testing purposes, formats them into documents, and so on.

- **Integration view**. This view is the view to the integration stream or branch, which shows the components and elements from all team members.

Every view has a view storage directory. ClearCase uses this directory to keep track of information such as which files are loaded into your view (for snapshot views only) and which versions are checked out to it. The view storage directory is for ClearCase administrative purposes only. Do not modify anything in it.

Naming Your Views

Each view has a unique, descriptive name, called a *view tag*. A view tag must be a simple name containing no special characters. While spaces are legal, we do not encourage view tags that contain spaces.

We recommend you choose a name that helps you determine the owner and purpose of the view. Names like **myview** or **work** do not describe the view's owner or contents; if you work with more than one view, such generic names can lead to confusion.

The following are examples of good view tags:

- susan_rel2
- laura_rel1_bugfix
- david_rel2_CR002311

Types of Views

ClearCase provides two types of views: dynamic views and snapshot views.

Dynamic views use a ClearCase file system, the MVFS (multiversion file system), to provide transparent access to the elements in the VOBs. When a new element version is created, you see it immediately in a dynamic view. The MVFS is available only if the MVFS option was selected during installation.

A snapshot view copies files from the VOBs (versioned object bases) onto your computer. You must update snapshot views frequently to make sure these views contain the versions specified by the stream (in UCM) or config spec (in base ClearCase).

ClearCase offers both dynamic views and snapshot views.

ClearCase LT offers only snapshot views.

Which View to Create

Although dynamic views offer an advantage over snapshot views by always showing the most recent version of an element, there are situations where snapshot views may be preferable.

The following sections identify some of the features that guide you in determining which type of views to create.

Considerations for Creating Dynamic Views

Create a dynamic view when:

- You do not have enough disk space to store view-private files or view-private directories.
- You start the view and mount VOBs on your computer.
- For any element in a mounted VOB, the view is always to select the version specified by the config spec. You can navigate the view-extended namespace to see any version of any element in any mounted VOB.
- It is acceptable that the checkout operation checks out the latest version on a branch, regardless of what the config spec specifies.
- It is important that you can use the clearmake and omake features of build auditing and build avoidance.
- The view objects need to be accessible to other team members through the ClearCase view-extended namespace.

Considerations for Creating Snapshot Views

Create a snapshot view when:

- You have enough disk space on your computer to store the copies of the loaded elements and any view-private files or view-private directories.
- Your work requires you to be frequently away from your office.
- Your view's config spec specifies which element versions to load into the view.
- It is acceptable to frequently update your view to ensure that you have the latest versions loaded into the view.

Preparing for Creating a View

To enable you to create a view in this tutorial, you need access to a VOB. The following steps guide you in creating a VOB on your computer so that you can then practice creating a view.

Follow the directions for the product you use:

- Creating a VOB Using ClearCase
- Creating a VOB Using ClearCase LT

Creating a VOB Using ClearCase

Create a local VOB as follows:

1. Start the View Creation Wizard:

   Click **Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB**

2. Specify the following:

   VOB name: *your-username*_**viewsmod_vob**

   Keep the option **VOB will contain UCM components** selected.

   Click **Next**.

3. Deselect the options Create VOB as a single-level component and Create as a UCM Project VOB.

   Click **Next**.

4. Click **Advanced Options**, select **Use Explicit Path**, and then click **Browse**.

5. Navigate to the cc-tut folder on your computer and click OK. The path now is similar to the following:
   \\computer-name**cc-tut***your-username*_**viewsmod_vob.vws**.

6. Click **OK**.

7. As administrative VOB to be associated with the view, click **none**.

8. Click **Finish.**

9. Click **OK**.

You now have a VOB for practicing the exercises of this module.

Creating a VOB Using ClearCase LT

Create a local VOB as follows:

1 Start the View Creation Wizard:

Click **Start > Programs > Rational Software > Rational ClearCase > Administration > Create VOB**

2 Specify the following:

VOB name: *your-username*_**viewsmod_vob**

Click **Next**.

3 Accept the option **Allow VOB to contain multiple components**.

Click **Next**.

4 Click **Finish.**

5 In the Confirmation dialog box, click **OK**.

6 In the Creating VOB dialog box, click **Close**.

You now have a VOB for practicing the exercises of this module.

Creating a View

The View Creation Wizard guides you in creating a dynamic or snapshot view.

To create a view, one or more VOBs need to be in place.

Follow of the instructions for the platform of interest to you:

- Creating a View on Windows
- Creating a View on UNIX

Creating a View on Windows

Starting the View Creation Wizard

The two most likely places from which you start the View Creation Wizard are the **ClearCase Explorer** and the **Start** menu.

From ClearCase Explorer

1 In the Shortcut window (left) of the ClearCase Explorer, click the **Toolbox** tab.

2 Click **Base ClearCase > Create View**.

From the Start Menu

Click **Start > Programs > Rational Software > Rational ClearCase > Create View**.

Try It!

1 Start the View Creation Wizard.

2 Continue by following the directions for the product you are using:

- Creating a View Using ClearCase
- Creating a View Using ClearCase LT

Creating a View Using ClearCase

Choosing the Project

You are going to create a snapshot view having the view tag *your-username*_**snapview** and it will be located in the cc-tut folder.

The first page of the View Creation Wizard prompts you to specify if the view you are creating is to be associated with a UCM project.

In the context of the tutorial, the answer is no.

Try It!

Complete the page as follows:

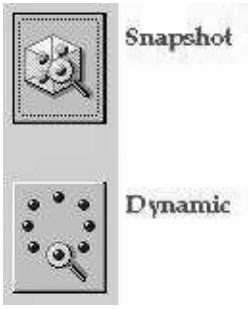
1 Select **No**.

2 Click **Next**.

Choosing the Type of View

The second page of the Create a View Wizard prompts you to choose the type of view to be created.

- Click the **Dynamic** or **Snapshot** button



Try It!

Complete this page as follows:

1. You are creating a snapshot view, so click the **Snapshot** button.
2. Click **Next**.

Specifying the Name of Your View

The third page of the Create a View Wizard prompts you to specify a view tag (name of your view) and a storage location.

The view tag for your view is: *your-username*_**snapview**.

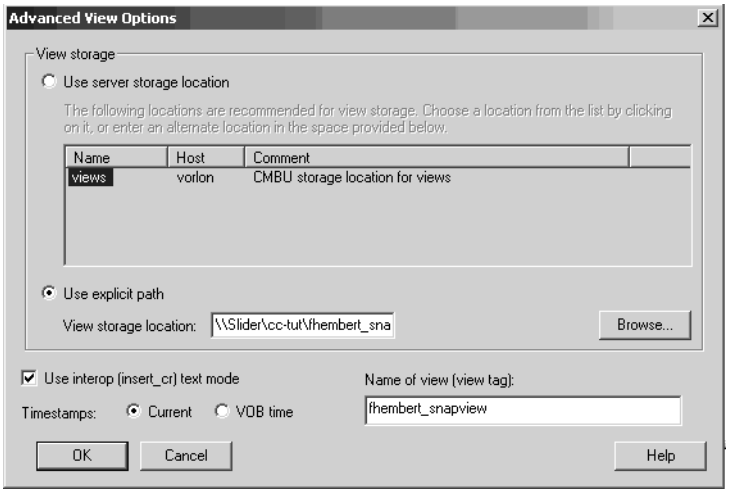
Try It!

Complete the page as follows:

1. Modify the default location for the view tag to point to the cc-tut folder so that it is similar to the following: **C:\cc-tut***your-username*_**snapview**.
2. Click the **Advanced Options** button.

Specifying the View Storage Location

The **Advanced Options** button lets you specify remote or local storage locations for the new view.



Let us examine the available options in more detail.

- View storage locations can be stored on a remote server or locally on your computer.

  To store the view on a remote host, select the **Use server storage location** option and select one of the available locations from the list.

  To store the view locally, select the **explicit path** option, and type the view storage location, or click the Browse button.

- The **Network region** option displays the name of the network region in which the view tag will be created. The network region is a logical subset of a local area network within which all hosts refer to VOB storage directories and view storage directories with the same network pathnames. The value shown in the box represents the default region to which the current host belongs. Do not change the network region unless you know that the server that will host the view is located in a different network region.

  The **Use interop (inser_cr) text mode** option lets you specify how lines in text files under ClearCase control are processed, independent of the platform on which the view storage, the VOB server, or the client is located. When selected, ClearCase inserts a <CR> character before every <LF> character to file elements whose element type is text_file or a subtype of text_file.

- The **Create shareable derived objects**, when selected (by default), enables clearmake and omake builds in this view to create derived objects that other team members can access using the **cleartool winkin** command. The **winkin** command

accesses one or more derived objects from a dynamic view, or converts a nonshareable derived object to a shareable derived object.

Try It!

The view is to be located on your computer in the cc-tut folder.

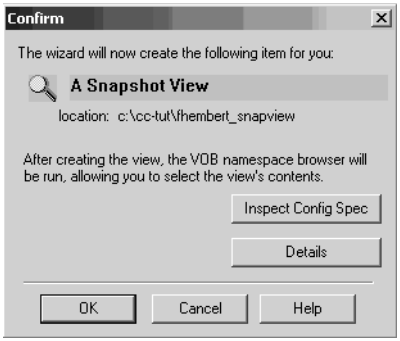
1 Verify that the storage location of the view is similar to the following:

Dynamic views**: C:***your-computer-name***\cc-tut***your-username***_snapview**

2 Verify the options **Use interop text mode** and **Create shareable derived objects** are selected.

3 Click **OK** to close the dialog box.

4 Click **Finish**.



5 On the Confirm dialog box, click **OK**.

ClearCase creates the view and then opens the Available Elements dialog box which lists the available VOBs.

6 Select the VOB *your-username***_viewsmod_vob** and click **Add**.

7 Click **OK**.

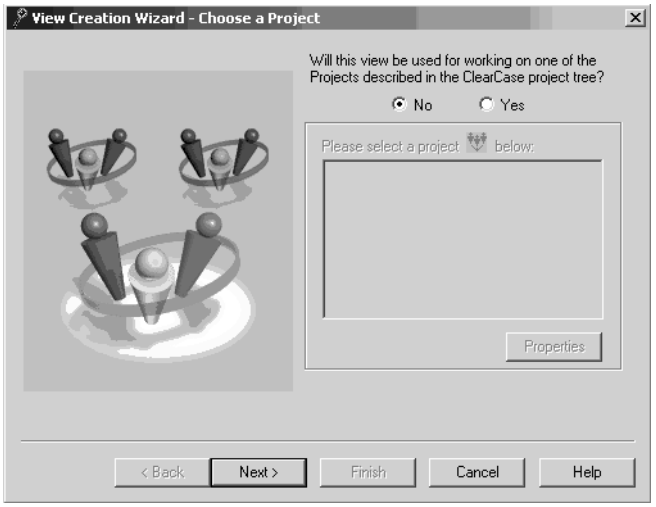
ClearCase starts copying the elements in the selected VOB onto your computer. The VOB used in this tutorial is empty and, therefore, no files are being copied.

Creating a View Using ClearCase LT

Choosing the Project

You are going to create a snapshot view having the view tag *your-username***_snapview** and it will be located in the cc-tut folder.

The first page of the View Creation Wizard prompts you to specify if the view you are creating is to be associated with a UCM project.



Try It!

In the context of the tutorial, the answer is no. Complete the page as follows:

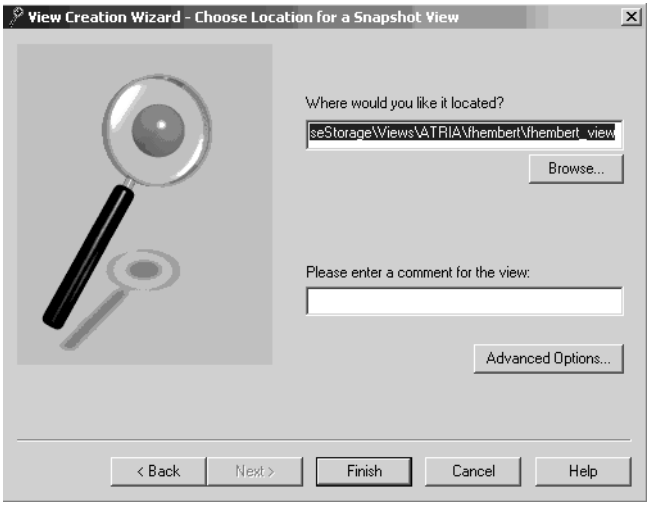
1 Select **No**.

2 Click **Next**.

Specifying the View Storage Location

In this page, ClearCase LT suggests a default storage location for your view.

The **Browse** button lets you specify a different storage location for the new view.

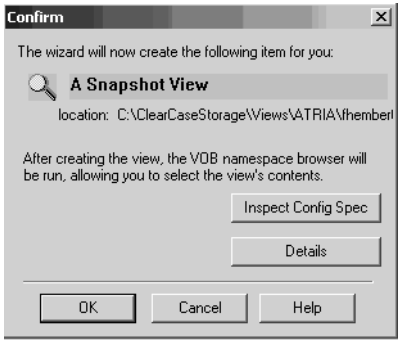


Try It!

The view is to be located on your computer in **C:\ClearCaseStorage\Views***domain**your-username**your-username*_**view.vws**

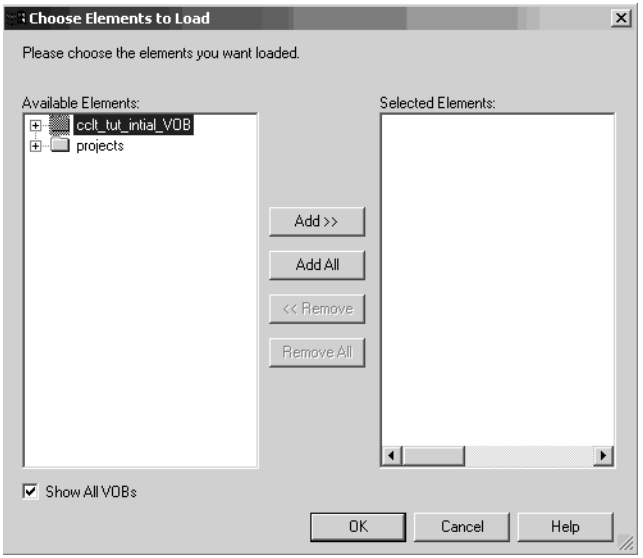
1 Verify that the storage location of the view is the same as indicated above. If not, click Browse and navigate to that location.

2 Click **Finish**.



3 On the Confirm dialog box, click **OK**.

ClearCase creates the view and then prompts you to identify the VOBs containing the elements to be loaded in the view.



4 Select the VOB *your-username*_**viewsmod_vob** and click **Add**.

5 Click **OK**.

ClearCase starts copying the elements in the selected VOB onto your computer. The VOB used in this tutorial is empty and, therefore, no files are being copied.

Creating a View on UNIX

When using UCM, the easier way to create a view is to use the Create View dialog box, which can be opened by selecting **Create View** in the Project Explorer, or by typing the **clearprojexp &** command at the system prompt in a terminal window. The VOB to which you want to set a view must be a UCM PVOB.

If the VOB is not a UCM PVOB, you cannot use the Create View dialog box, and you must use the **cleartool mkview** command.

The **mkview** command by default creates a dynamic view. To create a snapshot view you use the **-snapshot** option.

After you create a dynamic view, you must use the **cleartool setview** command to set the view to an existing VOB. This does not apply to snapshot views. The **setview** command creates a process that is set to the specified dynamic view. The new process is setting a context for the view.

Try It!

You are going to create a dynamic view named **dynview**.

1 Create the view by typing this command:

ct mkview -tag dynview /var/tmp/*your-username***_dynview.vws**

The system's response is similar to the following:

```
Created View
Host-local pth: stellar:/var/tmp/dynview.vws
Global path: /net/stellar/var/tmp/dynview.vws
It has the following rights:
User: fhembert : rwx
Group: user: rwx
Other: r-x
```

2 Set the view with the setview command as follows:

ct setview dynview

Starting a View

Follow the directions for the platform of interest to you:

- Starting a View on Windows
- Starting a View on UNIX

Starting a View on Windows

1 From the Shortcut pane in the ClearCase Explorer, click **Toolbox > Base ClearCase > Start View**.

The **Local** tab lists dynamic views for which the view-storage directory is located on your computer.

The **Recent** tab lists the dynamic views you have started recently.

The **All** tab lists the views registered in the current network region.

2 Click one of the tabs from which you will select the view.

3 Optionally, select **Restart at Logon** for the selected view to start every time you log on to your computer. The option is not available if you select multiple views.

If you do not select this option, you must restart the view if you want to use it after you restart your computer.

4 The option **Connect to Drive** assigns the view to a drive letter. In addition to any drive letter you select, all started views are accessible from the dynamic views drive. The option is not available if you select multiple views.

Starting a View on UNIX

The **cleartool startview** command enables processes on the local host to access a dynamic view. To start a view, type this command:

cleartool startview *view-tag*

Try It!

Start the view2modviews view by typing this command:

ct startview dynview

Stopping a View

ClearCase stops dynamic views when you log off. You can stop a dynamic view when you no longer want to use it or if you need to reclaim system resources used by the view.

Follow the instructions for the platform of interest to you:

- Stopping a View on Windows
- Stopping a View on UNIX

Stopping a View on Windows

To stop a view:

1. From the **Shortcut** pane in the ClearCase Explorer, right-click the view shortcut and select **View Properties**.
2. In the View Properties window, click **Stop View** on the **General** tab.

If the dynamic view is assigned to a drive letter, the view restarts when you log on, unless you first unassign the drive letter.

Stopping a View on UNIX

To stop a view, type this command:

cleartool endview *view-tag*

Try It!

Stop the **dynview** view by typing this command:

ct endview dynview

Updating a Snapshot View

When you use a snapshot view, you must perform frequent update operations to ensure that your view contains the latest versions of the elements.

Update Options

Before performing the update, you are able to see a preview.

When updating the entire view, you have these default options:

- **Update mode**. Evaluates the config spec against all the files and directories in the snapshot view and does the following:
  - If a version selected by the config spec is different from the version in the view, the Update Tool copies the version from the VOB to the view, overwriting the existing version.
  - Removes any elements no longer selected by the view.
  - Operates on hijacked files according to the options you set on the **Advanced** tab of the Start Update dialog box.
- **Leave hijacked files in place**. A hijacked file is a file that was modified in a snapshot view by manually changing the file's read-only attribute instead of checking out the file before modifying it. This option treats hijacked files as checked-out files: the version in the VOB is not copied to the view and the hijacked file is not updated. After the update operation is complete, you can choose to check out the hijacked files.
- **Set file times to current time**. For any file that is copied to the view during the update operation, ClearCase sets the file's time stamp to the current time on your machine.

You modify the defaults on the **Advanced** tab of the **Start Update** dialog box.

Updating the Contents of a Snapshot View

When you use snapshot views, you must update the contents of your views frequently to ensure that your view sees the most up-to-date versions of the elements in the VOB.

On Windows, you use **Update View** from the ClearCase Explorer.

On UNIX, you use the **cleartool update** command.

Follow the instructions for the platform of interest to you:

- Updating the Contents of a Snapshot View on Windows

- Updating the Contents of a Snapshot View on UNIX

Updating the Contents of a Snapshot View on Windows

1 From the ClearCase Explorer, right-click the root directory of the snapshot view and click **Update View**.

2 In the **Start Update** dialog box, you can do the following:

- To preview the update without actually performing the update, select the corresponding option. When the preview completes, you can perform the update.
- Click the **Advanced tab** and change default options for the update tool.

Updating the Contents of a Snapshot View on UNIX

You use the **cleartool update** to update the elements in your snapshot view directory. The command specifies the view path and not the view tag.

Try It!

This exercise guides you in creating a snapshot view **snapview** and in updating it with the **cleartool update** command.

1 Create the snapshot view **snap** by typing this command:

ct mkview -snapshot -tag snapview /var/tmp/your-username_snapview.vws

2 Update the snapshot view by typing this command:

ct update /var/tmp/your-usename_snapview.vws

The system's response is similar to the following:

```
Log has been written to
"/var/tmp/fhembert_snapview.vws/update.13-Dec-02.12:33:50.updt".
```

By default, ClearCase records any update information in a log file located in the directory of your snapview view. Each update is recorded in a separate log file that lists the date and time the update occurred.

3 View the content of the update log file by typing this command:

more /var/tmp/*your-username***_snapview.vws/update.***date-and-time***.updt**

Listing Active Views

Follow the instructions for the platform of interest to you:

- Listing Active Views on Windows
- Listing Active Views on UNIX

Listing Active Views on Windows

The Views tab contains all your views.

View > Refresh View Shortcuts updates your views and removes any view icons for views which no longer are associated with a VOB.

Listing Active Views on UNIX

You can easily identify active views by using the **cleartool lsview** command. In the system's response, active views have an asterisk to the left of the view tag.

The views you created in this module, or throughout the tutorial, are located in the **/var/tmp/** directory on your local system.

To view all views on your local system, type this command:

ct lsview -host *your-computer-name*

For example: **ct lsview -host goose**

The system's response is similar to the following:

```
dynview /var/tmp/fhembert_dynview.vws
* snapview /var/tmp/fhembert_snapview.vws/.view.stg
```

The asterisk next to **snapview** indicates that the view is still active.

Use the **cleartool endview** command to stop the view.

Removing Views

Follow the instructions for the platform of interest to you:

- Removing Views on Windows
- Removing Views on UNIX

Removing Views on Windows

With **Remove View** you remove views you no longer need.

To remove a view, right-click the view's icon on the **Views** tab and select **Remove View** from the shortcut menu.

Try It!

Remove the *your-username*_**snapview** view by right-clicking its icon on the **Views** tab and selecting **Remove View** from the shortcut menu.

Removing Views on UNIX

You remove views by using the **cleartool rmview** command specifying the view's path name as parameter and not the view tag

Before removing a view, you must first stop any active views using the **cleartool endview** command.

Try It!

During the exercises in this module, you created a dynamic view named **dynview** and a snapshot view named **snapview**. If you did the exercises as directed, you stopped the dynview view, but not your snapview.

During this exercise you will stop the **snapview** process and then remove the two views.

1. Stop the snapshot view by typing this command:

   ct endview snapview

2. Remove the **snapview** view using the **rmview** command as follows:

   ct rmview /var/tmp/*your-username*_**snapview.vws**

3. Now remove the **dynview** view by typing this command:

   ct rmview /var/tmp/*your-username*_**dynview.vws**

Controlling the Contents of Views

14

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	Base ClearCase
Products	ClearCase, ClearCase LT

A view is the mechanism ClearCase uses to provide access to a specific version of files and directories under source control. Rules determine which version of each element is visible and accessible through the view.

The set of rules that determines which version of an element to display is referred to as the view's config spec.

This module covers the following topics:

- How a config spec Works
- Understanding the Default config spec
- Understanding the Role of the config spec
- Understanding Include Files
- Changing the config spec

How a config spec Works

Snapshot views use config specs that contains two kinds of rules: load rules and version-selection rules.

Dynamic views use version-selection rules only and ignore any load rules.

Load rules specify which elements are copied into the view.

Version-selection rules specify the version that is visible in the view.

Figure 3 Config Spec on Windows

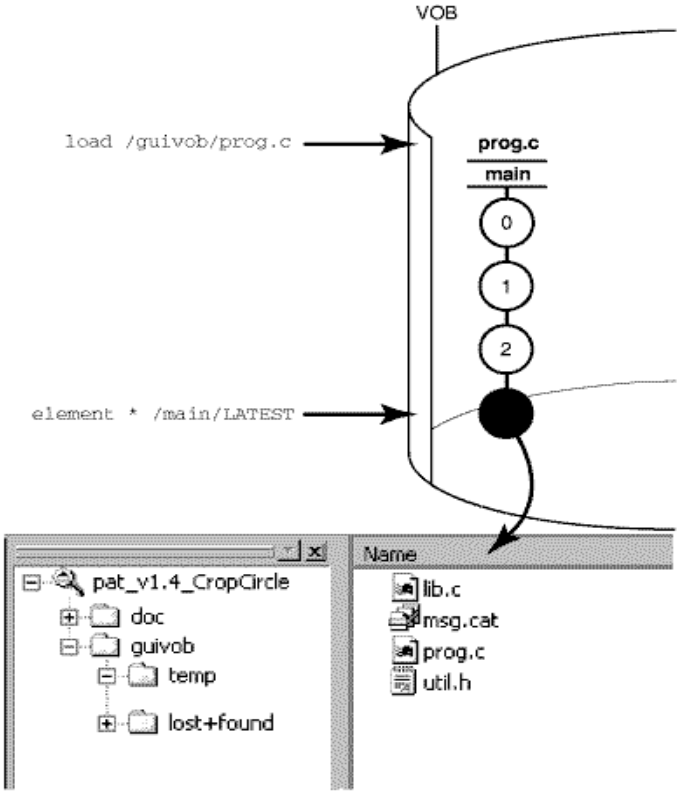


Figure 3 illustrates how ClearCase interprets the load rules to select the version of the file that is visible through your snapshot view.

Figure 4 Config Spec on UNIX

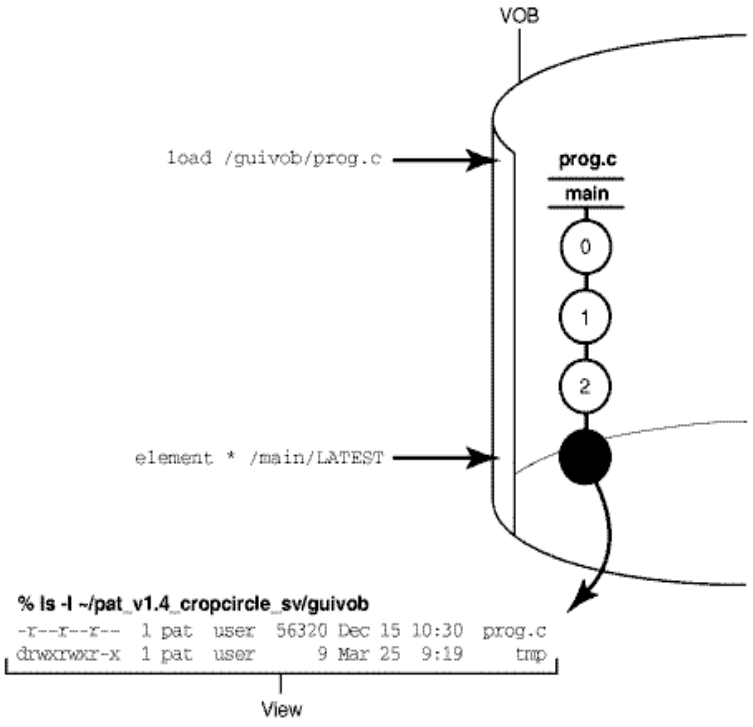


Figure 4 illustrates how ClearCase interprets the load rules to select the version of the file that is visible in your view.

For each element, the config spec attempts to locate a version that matches a config spec rule. If no version of the element matches any of the rules, the element is not accessible through the view.

A view can have only one config spec.

Understanding the Role of the config spec

Config specs are used to achieve a degree of control over project work.

Project managers use config specs for any of the following reasons:

- To control which versions developers see and what operations developers can perform in specific views.

- To prohibit checkouts of all selected versions or restrict checkouts to specific branches.

Understanding the Default config spec

The default config spec defines a dynamic configuration which selects checked-out or the latest version of every element on the main branch throughout the entire source tree by any developer.

The following two lines are the default config spec rules:

element * CHECKEDOUT
element * /main/LATEST

The first rule selects the checked-out version of the element in your view.

The second rule selects the latest version of that element on the **main** branch.

If another team member checks in a new version of an element on the **main** branch, the new and latest version appears immediately if your view is a dynamic view, and you do not have the element checked out.

With a snapshot view, the new version on the **main** branch becomes visible when you update your view.

Understanding Include Files

ClearCase supports an include file facility, which makes it easy to ensure that all team members use the same config spec.

Include files contain config spec rules and are placed in a public location. Subsequently, team members replace their default config spec with a one-line rule that points to the include file.

Include files are identified by the .csp extension; however, include files are not required to have the .csp file extension.

For example, the config spec to be used by all team members is called major.csp and is located in a publicly accessible location **/public/c_spec/**. Each team member replaces the content of their config spec with the following line:

Windows	**include \\main_svr\public\c_spec**major.csp
UNIX	**include /public/c_spec/**major.csp

If you create config specs to be shared between UNIX and Windows computers, where VOB tags are different, you must have two includes files, or you must store the include file in a UNIX directory that is accessible from both platforms.

Changing the config spec

Changing the config spec only applies to load rules; not to version-selection rules.

Select the platform that interests you:

- Changing the config spec on Windows
- Changing the config spec on UNIX

Changing the config spec on Windows

To modify the config spec of your current view, do the following:

1. In the **ClearCase Explorer** shortcut pane, right-click the view and click **View Properties** from the shortcut menu.
2. In the view's Properties dialog box, click the **Config Spec** tab and then click **Edit**.
3. Make your changes to the config spec and click **OK**.

Changing the config spec on UNIX

To modify the config spec of your current view, do the following:

1. Open a shell and change to a directory in the view.
2. Open the view's config spec in your default editor by entering this command:

   cleartool edcs &
3. In your text editor, use the following syntax to create load rules:

   load *vob-tag/element-pathname [...]*

   For example, the rule **load /guivob** loads all files and directories in the VOB named **/guivob**. The rule **load /guivob/batch** only loads the **batch** directory recursively.
4. Save the config spec and exit the text editor.
5. In your shell, answer **Yes** at the ClearCase prompt for setting the config spec.

Checking Files In and Out

15

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	UCM, base ClearCase
Products	ClearCase, ClearCase LT

Checking files in and out are two basic operations you will perform on a regular basis when working with ClearCase.

ClearCase uses a check out > edit > check in model. Before you can check out a file, the file must exist and must be under source control.

This module covers the following topics:

- Checkin and Checkout Requirements
- Adding Files to Source Control
- About Reserved and Unreserved Checkouts
- Checking Out Files and Directories
- Finding Checked Out Files
- Canceling a Checkout
- Displaying a History of Checkouts
- About Hijacked Files
- When to Check In Files
- Checking In Files

Checkin and Checkout Requirements

To gain access to files to perform checkin and checkout operations, the following must be in place:

- The development environment must be configured and set up. Depending on whether your company has chosen to adopt the UCM process, VOBs or UCM

Projects should be in place, including the development and integration streams or branches.

- For existing projects, specific versions of project-related files are selected as the starting point for the next development effort. For new projects, the development infrastructure has been defined and set up.
- As an individual contributor in a project, you have mounted the VOBs that hold all project-related files, and you have created a view (dynamic or snapshot) to access such VOBs.

Adding Files to Source Control

Before you can check out a file, you must first add it to ClearCase source control.

Follow the instructions for the platform of interest to you:

- Adding Files to Source Control on Windows
- Adding Files to Source Control on UNIX

Adding Files to Source Control on Windows

To add a file to source control in the ClearCase Explorer, do the following:

1. In the Browser pane, select the private file that you want to add to source control.
2. Right-click and select **Add to source control from the shortcut menu.**
3. Type an activity and a comment, and click **OK**.

Try It!

Note: To have a file to use for learning to check in and out, the exercise includes creating a small text file named author.txt.

ClearCase users create the author.txt file in the VOB *your-username*_tut_elements_vob.vbs.

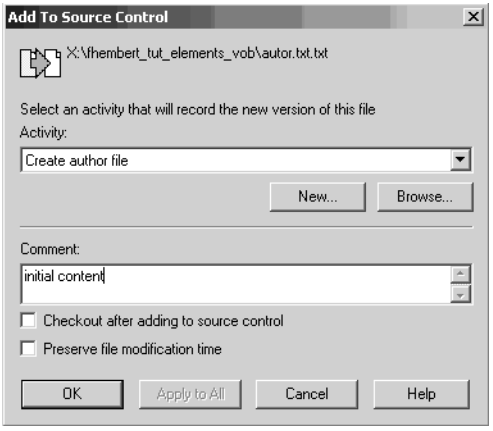
ClearCase LT users create the author.txt file in the VOB cclt_tutorial_sources.vbs.

1. Use your favorite text editor, and create a text file and type in your name.
2. Save the file using the name author.txt in the *your-username*_**tut_elements_vob** folder. Then close the editor.
3. Refresh your view (press F5) to see the file you created listed.

4 To add the author.txt file to source control, right-click the file and select **Add to source control from the shortcut menu.**

5 **ClearCase users only**. As activity, type: **author file**. As comment, type: **initial content**.

6 Make sure the option **Check out after adding to source control** is not selected, to enable you to practice checking out later. Then, click **OK**.



7 Go to the ClearCase Explorer and press F5 to update the contents of your view. ClearCase has added the author.txt file to source control.

Adding Files to Source Control on UNIX

Before you can create a new element, or add an existing element to source control, you must check out all parent directories of the element using the **cleartool checkout** command. To retain the parent directory of the view-private file that is being created as a new element, you use the **-mkpath** option in the **cleartool checkout** command.

You add a file to source control by using the **mkelem** command. This command creates one or more new elements. By default, this command will check out the element to your view.

If the element already exists as a view-private file, use the **-checkin** option to create the new element using the view-private file. To preserve the modification time of the file

being checked in, use the **-ptime** option. If you omit the **-ptime** option, the modification time of the new version is set to the checkin time.

For example, to add a view-private file named foo.c to source control that is located in the bin directory, you would do the following:

1 Checkout the bin directory:

cleartool checkout -mkpath bin

2 Add foo.c to source control and retain the file's modification time:

cleartool mkelem -checkin -ptime foo.c

You should not forget to check in the parent directories when you are done, using the **cleartool checkin** command!

Try It!

Note: To have a file to use for learning to check in and out, you will first create one in **/var/tmp/***your-username***_tut_elements_vob**.

1 Navigate to the VOB **/var/tmp/your-username_tut_elements_vob**.

2 Use your favorite text editor, and create a text file containing your name.

3 Save the file using the name author.txt. Then close the editor.

4 Check out the parent directory of the author.txt file by typing this command at a system prompt:

cleartool checkout *your-usename***_tut_elements_vob**

5 To add the view-private file author.txt to source control under, type this command:

cleartool mkelem -ci -ptime -co "initial content" author.txt

Optional. Review the reference page for the cleartool mkelem command and display it in a Web browser by typing this command: **cleartool man -gra mkelem**

About Reserved and Unreserved Checkouts

ClearCase lets you choose between reserved and unreserved checkouts. You specify which type of checkout you want during the checkout process.

A reserved checkout gives you the exclusive right to check in a new version of the element. When you check out a file reserved, only you can make changes to the file. No one else can check out the file reserved.

Team members can modify the content. They just have to check out the file unreserved.

With an unreserved checkout, you may need to merge your changes at checkin time, if someone else checked in a newer version before you did. When you check out a file or directory unreserved, the file remains available for a reserved checkout. Other users can check out the file or directory with a reserved or unreserved checkout.

Checking Out Files and Directories

Files and directories can be checked out from any of the following places:

- The ClearCase Explorer. (Windows only)
- My Computer and the Microsoft Windows Explorer. (Windows only)
- The Command Line. (Windows and UNIX)
- From applications that are integrated with ClearCase, such as, Visual InterDev, PowerBuilder, FrontPage, Word, and so on. ClearCase adds checkin and checkout menu items to the integrated applications' menu structure.

The tutorial focuses on checkouts made from the ClearCase Explorer on Windows and the command line on UNIX.

Follow the instructions for the platform of interest to you:

- Checking Out Files on Windows
- Checking Out Files on UNIX

Checking Out Files on Windows

You learned how to check in a file from the Windows Explorer. In this section, you will learn how to check out a file from the ClearCase Explorer.

1. Select the files you want to check out.
2. Right-click and select **Check Out** from the shortcut menu. You can check out multiple files at the same time.
3. Enter a comment.
4. If you searched for checkouts and know that the file is already checked out reserved, select **Unreserved if already reserved**, and click **OK**.

A white check mark in a green circle to the left of the filename indicates the file is checked out.

Try It!

This exercise requires that you performed the exercise in section *Adding Files to Source Control* earlier in this module.

1 Make the view current by clicking its icon in the shortcut pane on the **Views** tab:

- ClearCase users: *your-username*_**denali_release_development**
- ClearCase LT users: *your-username*_**snapview**

2 Right-click the file author.txt and select **Check Out** from the shortcut menu.

3 ClearCase users only: Add a new activity by clicking New and typing: **adding address**, and click **OK**.

4 Add an optional comment.

5 Verify that the **Reserved** option is selected, and then click **OK**.

ClearCase checks out the file; a white check mark in a green circle appears to the left of the checked-out file.

Checking Out Files on UNIX

From a command or terminal window, enter this command:

cleartool checkout -reserved -c "*comment" filename*

For example, you need to fix a typo in a text string that appears in a dialog box. A Quality Assurance engineer reported the problem in a defect report, number DEF0012345. The typo is in the file foo.c.

To check out the file foo.c from a command or terminal window, you would enter this command:

cleartool checkout -reserved -comment "Fix DEF0012345 - typo in text string" foo.c

Try It!

Check out the author.txt file, specifying an activity and adding a comment, by typing the following commands at the system prompt of a terminal window:

1 Navigate to **/var/tmp/***your-username***_tut_elements_vob.**

2 Type this command:

cleartool checkout -reserved -comment "Adding address" -activity "add_address" author.txt

ClearCase checks out the author.txt file.

Finding Checked Out Files

On Windows, the Find Checkouts tool lets you search for files that you and other coworkers have checked out.

On UNIX, you use the **cleartool lscheckout** to search for checked-out files.

Follow the instructions for the platform of interest to you:

- Searching for Files on Windows
- Searching for Files on UNIX

Searching for Files on Windows

Starting the Find Checkout Tool

You start the Find Checkout tool as follows:

1. Select a file in your view.
2. Right-click and select **Find Checkouts** from the shortcut menu. The Find Checkout window opens, and you are prompted to specify search criteria in the Find Criteria dialog box.
3. Set the your search criteria and click **OK**.

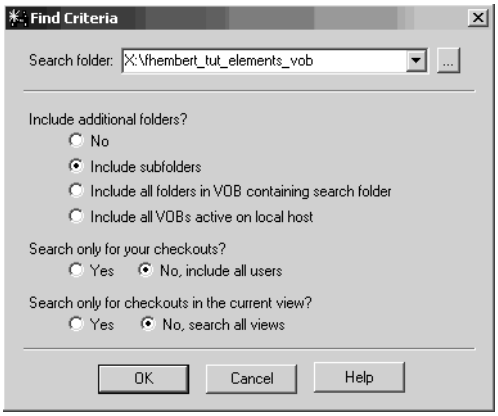
The result is displayed in the Find Checkout Window.

Try It!

Select the author.txt file, right-click and select **Find Checkouts**.

Understanding the Find Criteria Options

The various options in the Find Criteria dialog box enable you to specify search criteria that will help you find checked-out files.



Let's examine the various options.

Search Folder lists the directory where the selected file is located. You can change location by clicking the down arrow or the **More** button. Change the listed directory path if you started the Find Checkout tool from a random location in the view.

The options under **Include additional folders** let you to widen the search path to include subdirectories, and even other VOBs.

In **Search only your checkouts**, you can widen the search to include files checked out by other people. The default is to search for your checkouts only.

In **Search only for checkouts in the current view**, you can widen the search to include checkouts in all views. The default is to search in the current view only.

Try It!

1 In the **Find Criteria** dialog box do the following:

- By default, ClearCase automatically completed the **Search folder** box with the name of the *your-username*_**tut_elements_vob** folder. This is correct, so keep it.
- By default, the option **Include subfolder** is selected. This is correct, so keep it.
- Select the option **No, include all users**. In a real development environment, it is good practice to select this option to find out if another team member has checked out the file.
- Select the option **No, search all views**. This enables you to find out if you checked out the file from another view. Checking out a file from one view, and later checking in the same file but from a second view, creates a situation where the checkin of the file from the second view eclipses the file from the view from which it was checked out.

2 Click **OK** to start the search.

ClearCase finds the checked out author.txt file and lists it in the **Find Checkouts** window.

The Find Checkouts window provides other useful information about the checkout, such as:

- The username of the individual who checked out the file.
- The name of the view from where the file was checked out.
- Whether or not the checkout is reserved or unreserved.
- The preceding version of the file.
- The activity associated with the checkout.

3 Click **File > Exit** to close the Find Checkouts window.

4 Check out the author.txt file out again, and practice adding a new activity to be associated with the checkout.

Searching for Files on UNIX

On UNIX you use the **cleartool lscheckout** command from the system prompt.

Here are some examples of the **lscheckout** command formats you might find useful.

- **cleartool lscheckout -long -me**

  When including the **-long** option in the command, ClearCase includes the name of the view to which the element is checked out.

  With the **-me** option, ClearCase lists only those elements that you have checked out.

- **cleartool lscheckout -cview -user** *username* **-avobs**

  The **-cview** option narrows the search to elements checked out in the current view.

  The **-user** option lets you specify the username of a team member.

  The **-avobs** option specifies to search in all VOBs.

Try It!

1 Navigate to the **/var/tmp/***your-username***_tut_elements_vob** by typing this command:

cd **/var/tmp/***your-username***_tut_elements_vob**

2 Check out the author.txt file by typing this command:

cleartool checkout -comment "adding address" -activity "update_address" author.txt

3 **Optional**. Review the **lscheckout** reference page to learn about the other command options.

Canceling a Checkout

There are several reasons to cancel a checkout. Consider the following situations.

- You checked out the wrong file.
- The file is already checked out by another team member, and you would rather wait until your coworker checks in the file.

Follow the instructions for the platform of interest to you:

- Canceling a Checkout on Windows
- Canceling a Checkout on UNIX

Canceling a Checkout on Windows

You cancel a checkout from the ClearCase Explorer as follows:

1 Select the checked-out file.

2 Right-click and select **Undo Checkout** from the shortcut menu.

3 If you want to retain a private copy of the file, in the Confirm Undo Checkout dialog box, keep the option **Save copy of the file with a .keep** extension selected. You can remove the .keep extension later on by renaming the file.

4 Click **Yes** to cancel the checkout.

Try It!

1 Select the checked-out author.txt file.

2 Right-click and select **ClearCase > Undo Checkout** from the shortcut menu.

3 In the Confirm Undo Checkout dialog box, keep the option **Save copy of the file with a .keep extension** selected.

4 Click **Yes** to cancel the checkout.

5 Look for the copy of the author.txt file with a .keep extension: author.txt.keep. You may need to press F5 to refresh your view. Notice that this copy is not under

ClearCase source control and is a view-private file. You can rename the file to any name.

6 Check out the author.txt file again.

7 Modify the contents, by adding, for example, your street address.

8 Save the file and close the editor.

9 Check in the author.txt file.

ClearCase creates version 2 of the author.txt file on its main branch.

Canceling a Checkout on UNIX

From a command or terminal window, enter this command:

cleartool uncheckout -keep *filename*

For example, you need to fix a typo in a text string that appears in a dialog box. A Quality Assurance engineer reported the problem in a defect report, number DEF0012345. The typo is in the file foo.c.

To cancel the checkout of file foo.c from a command or terminal window, you would enter one of the following commands:

cleartool checkout -keep foo.c if you wanted to keep a private copy of the file.

cleartool checkout -rm foo.c if you did not want to keep a private copy of the file.

Try It!

To cancel the checkout of the author.txt file, type this command:

cleartool uncheckout -keep author.txt

Displaying a History of Checkouts

With each checkout and checkin, a new version of the file is created. Over a period of time, these checkouts and checkins create a change history, which you can view using the History Browser or the **lshistory** command.

Follow the instructions for the platform of interest to you:

- Displaying the History of Checkouts on Windows
- Displaying the History of Checkouts on UNIX

Displaying the History of Checkouts on Windows

To display the history of a file in the ClearCase Explorer:

1 Select the file whose history you want to examine.

2 Right-click and select **History** from the shortcut menu.

Try It!

1 Select the author.txt file, right-click and select **History** from the shortcut menu.

2 Review the information that ClearCase provides on the history of the author.txt file:

- The date when the file was modified.
- The username of the individual who modified the file.
- The name of the file.
- The type of action on the file; in this exercise, you see **create version** and you may also see **create branch.**
- The version number. Take note that the initial version number is 0, the second version is number 1, and so on.

Displaying the History of Checkouts on UNIX

To display a history of checkouts on UNIX, you use the **cleartool lshistory** command. Also, you must be in a VOB directory.

Here are some examples of the **lshistory** command formats you might find useful.

- **cleartool lshistory -long -me**

  When including the **-long** option in the command, ClearCase includes the name of the view.

  With the **-me** option, ClearCase lists only those elements on which you have worked.

- **cleartool lshistory -cview -user** *username*

  The **-cview** option narrows the listing to elements in the current view.

  The **-user** option lets you specify the username of a team member who has worked on elements in the current view.

Try It!

1 Navigate to the directory **/var/tmp/your-username_tut_elements_vob**

2 Type the command: **cleartool lshistory** author.txt

3 Type the command: **cleartool history**.

4 Observe how the system response is different when you do not specify a filename.

5 **Optional.** Review the **lshistory** reference page to learn about the other command options.

About Hijacked Files

When ClearCase loads a file element into a snapshot view, it applies the file system's read-only attribute to the file. If you change this attribute, and modify the file without checking it out first, ClearCase considers the file hijacked.

Only files under source control can be hijacked; view-private files are not under ClearCase control and you can modify them without involving ClearCase.

Hijacking takes a file outside of direct ClearCase control. Although ClearCase detects the change to the file, we recommend that you do not hijack files as standard practice.

You cannot hijack files in dynamic views.

Hijacking does not apply to directory elements.

Try It!

Because hijacking files can put you in a precarious situation, learn more about this subject by searching the Help for topics on hijacking files.

When to Check In Files

It is good practice to check in your files regularly, such as at the end of the week, or even at the end of the day.

While practices differ among development teams, even within the same company, it is good practice to check in your files when:

- You have made substantial changes to the checked-out file.
- Other team members need access to your file when, for example, they would like to perform a test build.
- When you foresee being out of the office.

Checking In Files

To check in one or more files, follow the instructions for the platform of interest to you:

- Checking In Files on Windows
- Checking In Files on UNIX

Checking In Files on Windows

1. Select the checked-out file.
2. Right-click and select **Check In** from the shortcut menu.
3. In the Check In dialog box, verify that the comment entered at checkout time is still meaningful. If not, replace it.
4. Click **OK**.

Try It!

1. Select the checked-out file author.txt.
2. Right-click and select **Check In** from the shortcut menu.
3. In the Check In dialog box, verify that the comment entered at checkout time is still meaningful.
4. Click **OK**.

Checking In Files on UNIX

From a command or terminal window, enter this command:

cleartool checkin -comment "*comment***"** *filename*

For example, after fixing a typo, you would enter one of the following commands to check in the corrected file:

cleartool checkin -comment "fixed typo reported in DEF0012345" foo.c

Try It!

To check in the file author.txt, type this command:

cleartool checkin -nc author.txt

Working with Baselines

16

Audience Project managers

Platforms Windows, UNIX

Process UCM

Products ClearCase, ClearCase LT

Achieving major milestones in a development project represents progress in reaching the final objective. At these points in time, you as project manager, can select stable versions of source files to build a version of the product that are then tested by the Quality Engineering team.

Quality engineers use each successful product build to install and test the product against the project requirements. When the tests are successful, you then define the successful product build to become the basis, or baseline, for achieving the next major milestone.

With each new approved baseline, developers update their private work environment by populating it with the sources of the approved build. This update operation is called rebasing.

This module covers the following topics:

- What Is a Baseline?
- Types of Baselines
- Comparing the Contents of Two Baselines
- Creating a Baseline
- Default Promotion Levels
- Promoting a Baseline
- Recommending a Baseline

What Is a Baseline?

A baseline represents a stable configuration for a set of files and directories. A baseline identifies activities and one version of every element visible in one or more components.

A new baseline includes the work developers have delivered to the integration stream since the last baseline. By making new baselines regularly, you ensure that developers stay in sync with each other's work.

Types of Baselines

You can create two types of baselines: *full* and *incremental*.

A full baseline is a baseline that you create by recording all versions of all elements below the component's root directory.

An incremental baseline is a baseline that you create by recording the last full baseline and those versions of elements that have changed since the last full baseline was created.

Generally, it takes less time to create an incremental baseline. However, it takes less time to look up the contents of a full baseline.

Comparing the Contents of Two Baselines

You can compare baselines to identify the changes made from one baseline to the next. Note that comparing baselines can only be done on nonbinary sources.

Follow the instructions for the platform of interest to you:

- Comparing Baselines on Windows
- Comparing Baselines on UNIX

Comparing Baselines on Windows

To compare two baselines, do the following:

1. In the component tree diagram, select a baseline.
2. Click **Tools** and select of the following:
   - **Compare with Another Baseline**. Compares the selected baseline with an earlier baseline.

- **Compare with Previous Baseline**. Compares the selected baseline with its immediate predecessor

3 If you select **Comparing with Another Baseline**, click the baseline you want to compare with the first selected one.

Differences in content are displayed in the **Compare Differences** dialog box.

Comparing Baselines on UNIX

The **cleardiffbl** command lets you display the differences between two baselines in the diffbl browser.

The command format is of the following form:

- **cleardiffbl** *baseline-selector1 baseline-selector2*

For example:

cleardiffbl OM_proj7.0_Integration_10_01_02 OM_proj7.0_Integration_04_07_02

This command shown in the example opens the diffbl browser and displays the differences between the baselines of October 1, 2002 and April 7, 2002.

Creating a Baseline

When you are ready to create a baseline, you must ensure that you are working in a static environment, namely, where developers cannot deliver their work to the shared work area.

Creating a Baseline on Windows

You create a static environment by first locking the integration stream.

Locking the Integration Stream

To lock the integration stream:

1 In the Project Explorer, select the project's integration stream.

2 Right-click and select Properties.

3 In the Properties dialog box:

a Click the **Lock** tab, and select Locked.

b Type an optional comment

c You can allow individuals to deliver to the integration stream by typing their usernames in the **Excluded Users** box. However, when your intention is to create a new baseline, you should refrain from excluding any developers from delivering to the integration stream.

d Click **OK**.

Creating the New Baseline

The integration being locked and unavailable to developers, you can now create a new baseline.

4 Click **Tools > Make Baseline** to open the **Make Baseline** dialog box.

5 Optionally, replace the default name in the **Baseline Root Name** box by typing a name that is meaningful to your project. By default, ClearCase names baselines by appending the date to the project name. For example: OM_proj7.0_Integration_10_01_02.

6 Next, choose the type of baseline to create: **incremental** or **full**.

7 Select a view context. The view must be attached to the stream in which you are making the baseline.

8 Click **OK**.

After you create the baseline, you should communicate the availability of the baseline to the developers so that they can update, or rebase, their private work environment.

You should also unlock the integration stream, so that developers can resume delivering their work to the integration stream in the shared work area after rebasing their private work areas.

Unlocking the Integration Stream

To unlock the integration stream:

1 In the Project Explorer, select the project's integration stream.

2 Right-click and select Properties.

3 In the Properties dialog box:

a Click the Lock tab, and select Unlocked.

b Type an optional comment

c Click **OK**.

Default Promotion Levels

ClearCase provides these five default promotion levels:

- REJECTED
- INITIAL
- BUILT
- TESTED
- RELEASED

You can replace these default promotion levels with your own, or add new ones.

Promoting a Baseline

As work on your project progresses, and the quality and stability of the source files improves, you change the baseline's promotion level to reflect a level of testing that the baseline has passed.

In UCM projects, a promotion level is a baseline property that can be used to indicate the quality or degree of completeness of the activities and versions represented by that baseline.

To promote a baseline, do the following:

1. In the Project Explorer, select the integration stream.
2. Click **File > Properties** to open the integration stream's property sheet.
3. Click the **Baselines** tab and continue as follows:
   a. In the **Components** box, select the component that contains the baseline you want to promote.
   b. In the **Baselines** box, select the baseline.
   c. Click **Properties** to display the baseline's property sheet.
   d. Select the new promotion level from the **Promotion Level** list.
   e. Click **OK**.
4. Back on the **Baselines** tab, you can change the promotion level of another baseline by repeating steps a through e.
5. Click **OK**.

After you promote the baseline, your next step is to recommend the baseline.

Recommending a Baseline

Recommended baselines are the set of baselines that project team members will use to update, or rebase, their development streams. When developers join a UCM project, their development work areas are initialized with the recommended baselines.

To recommend a baseline, do the following:

1. In the Project Explorer, select the integration stream.
2. Click **Tools > Recommended Baselines**.
3. In the **Recommended Baselines** dialog box, click **Add**.
4. In the **Add Baseline** dialog box, select the baseline to be recommended.

Working with Branches

17

Audience	Project managers, developers
Platforms	Windows, UNIX
Process	Base ClearCase
Products	ClearCase, ClearCase LT

Every element in ClearCase has, by default, a **main** branch.

If you use the default config spec, each time you check out a file to modify it, and then check it back in, you create a new version of that file on its **main** branch in the VOB. If you are not using the default config spec, the new version is created on the branch specified by your config spec.

The different element versions under version control are organized in a version tree. To facilitate the development effort of large projects, you can work in parallel on subbranches off the **main** branch.

This module covers the following topics:

- What Is a Branch?
- When to Use Branches
- The Config Spec-Views Connection
- Creating a Branch

What Is a Branch?

ClearCase uses branches to organize the different versions of files, directories and objects that are placed under version control. A branch is an object that specifies a linear sequence of versions of an element.

The entire set of an element's versions is called a version tree. By default, ClearCase provides for every single element in a VOB one principal branch, called the **main** branch. This **main** branch may also have subbranches.

A branch type is a type object that is used to identify a parallel line of development for an element. Each branch in an element's version tree is an instance of a branch type that exists in that element's VOB.

When to Use Branches

A branch represents an independent line of development. Typically, different kinds of work is done on different branches.

For example, to separate the work that is related to defect fixing from the regular development work, you can create a separate branch for the defect fixing of the **main** branch. The team working on the defect fixes can do their work without affecting or being affected by the work being done on the development branch.

When the work on the subbranch is completed, you can integrate onto the **main** branch. This is done by merging the most up-to-date version on the subbranch onto the elements' **main** branch.

The decision to use multiple branches for your project is part of a careful development planning strategy.

The Config Spec-Views Connection

The rules in a view's config spec determine which versions are visible in the view, and for snapshot views, which elements are loaded in the view.

We will first examine the rules in the default config spec. Then, by using an example, we will show how the rules in the config spec select specific versions.

The Default Config Spec Rules

The default config spec defines a dynamic configuration which selects checked-out or the latest version of every element on the **main** branch throughout the entire source tree by any developer.

The following two lines are the default config spec rules:

element * CHECKEDOUT
element * /main/LATEST

The first rule selects checked-out version of the element in your view.

The second rule selects the latest version of that element on the **main** branch.

If another team member checks in a new version of an element on the **main** branch, the new and latest version appears immediately if your view is a dynamic view, and you do not have the element checked out.

With a snapshot view, the new version on the **main** branch becomes visible when you update your view.

Config Spec Rules for Elements in Subbranches

The following is an example of a config spec that isolates work from the **main** branch by creating a subbranch called **bug-fix**:

```
element * CHECKEDOUT
element * .../bug-fix/LATEST
element * BASELINE-X -mkbranch bug-fix
element * /main/0 -mkbranch bug-fix
```

Rule 1 (first line) selects checked-out versions in your view.

Rule 2 (second line) selects the latest version on the **bug-fix** branch.

Rule 3 (third line) selects the version labeled **BASELINE-X**. When such a version is checked out, the **bug-fix** branch is created.

Rule 4 (fourth line) determines that when a new element is created, the **mkbranch** clause causes the new element to be checked out on the **bug-fix** subbranch which is created automatically with the **mkbranch** command. Also, if an existing element does not have a **bug-fix** branch or a version with the **BASELINE-X** label, the bug-fix branch is created when you check out the element.

Include Files for Config Specs

ClearCase supports an include file, which makes it easy to ensure that all team members use the same config spec.

Include files contain config spec rules and are placed in a public location. Subsequently, team members replace their default config spec with a one-line rule that points to the include file.

Include files are sometimes identified by the .csp extension. However, include file are not required to have the .csp file extension.

For example, the config spec to be used by all team members is called major.csp and is located in a publicly accessible location /**public**/**c_spec**/. Each team member replaces the content of their config spec with the following line:

Windows	**include \\main_svr\public\c_spec**\major.csp

UNIX	**include /public/c_spec/**major.csp

If you are creating config specs to be shared between UNIX and Windows computers, where VOB-tags are different, you must have two sources, or you must store the config spec in a UNIX directory that is accessible from both platforms.

Creating a Branch

Creating a branch involves the following two operations:

1. Create a branch type to enable the instantiations of branches. You do this by running the **mkbrtype** command.
2. Instantiate a branch. To do so, you use the **mkbranch** command.

The next two sections cover these two operations.

Creating a Branch Type Object (Windows and UNIX)

To create a branch type object, you use the **cleartool mkbrtype** command.

The following examples show the use of the **mkbrtype** command to perform the tasks as explained in the examples.

- **cleartool mkbrtype -c "bugfix development branch for V1" bugfix_v1**

  This command creates a branch type object named **bugfix_v1** which can be used only once in an element's version tree. The **-c** option allows for entering the comment "bugfix development branch for V1"

- **cleartool mbrtype -nc -pbranch patch2 rel2_bugfix**

  This command creates two branch types: **patch2** and **rel2_bugfix**. The **-nc** option indicates that you do not wish to provide a comment. The **patch2** branch is used to contain the program patches; the **rel2_bugfix** branch is used for bug fixing.

Instantiating a Branch (Windows and UNIX)

After you have created the branch type objects, you can create branches of that branch type. You do this by running the **mkbranch** command. The **mkbranch** command creates a new branch in the version trees of one or more elements. By default, the branch is checked out unless you use the **-nco** (no checkout) option.

In most cases, branches are created when you check out files or directories. The mkbranch rules in your config spec specify when new branches should be created.

The following examples show the use of the **mkbranch** command to perform the tasks as explained in the examples.

- **mkbranch -nc rel2_bugfix** hello.c

  This command creates a branch of the type **rel2_bugfix** off the version of hello.c in the view and checks out the initial version on the branch.

- **mkbranch -version \main\1 -nco -nc maintenance** util.c

  This command creates a branch of the type **maintenance** off version 1 off the **main** branch **of** the file util.c**,** and does not check out the branch.

Delivering Your Work 18

Audience	Developers
Platforms	Windows, UNIX
Process	UCM
Products	ClearCase, ClearCase LT

After testing your work in your private workspace, and you are sure that your work is error-free, you are ready to submit your work to the integration stream so that other project members can use the latest version of your work. In ClearCase, you do so by performing a deliver operation.

This module covers the following topics:

- Overview of the Deliver Process
- Performing a Deliver Operation

Overview of the Deliver Process

When you perform a deliver operation, the deliver process takes you through the following phases:

- Phase 1: Preparation and Preview
- Phase 2: Starting to Deliver
- Phase 3: Merging Differences
- Phase 4: Testing Your Work
- Phase 5: Completing the Delivery

In the following sections, we will examine these phases in more detail.

The information provided is applicable to the Windows and UNIX platforms.

Performing a Deliver Operation

The following sections guide you through the various phases of a deliver operation.

Phase 1: Preparation and Preview

In this phase you prepare your work area and **preview** the activities to be delivered. Activities represent units of work you have completed. You start by first listing the activities in your development stream that represent work that has yet to be delivered to the integration stream.

You prepare your work area as follows:

1. If your project manager has created a new recommended baseline since you last rebased, you must first rebase your development area.

   Rebasing your work area is a multistep process similar to the deliver process described in detail in the module *Rebasing Your Private Work Area*.

2. If your integration view is a snapshot view, you must first update it and resolve any hijacked files.

   Hijacked files are files in your view that were under ClearCase control before you changed the file's read-only attribute and modified the file without first checking it out. Hijacking files can only occur when working in snapshot views. Because hijacking files creates a potential for conflicts, we strongly recommend that you first check out the files you need to modify.

   After a file is hijacked, it is no longer under ClearCase control. However, ClearCase detects the hijacking because the file was under its control at one point in time.

   To update your snapshot view, you use the Update tool.

   - On Windows, you can start the Update tool from the ClearCase Explorer, by selecting the root directory of your snapshot view, right-clicking, and then selecting Update View from the shortcut menu.
   - On UNIX, you can use the update command to start the Update tool:

     cleartool update -graphical

     This command applies only to dynamic views.

   The Update tool enables you to preview an update without actually performing the update.

   After you have set the options in the Update tool, click **OK** to start the update operation.

ClearCase begins the update operation and displays a progress indicator. When the update is complete, the Snapshot View Update window displays a categorized list of the actions taken to update the view.

3 Lastly, find, compare, and check in the work you want to deliver.

At this point, if there are any hijacked files, you need to resolve any issues related to the hijacking of the files.

Also, you need to make sure that all the files ready for delivering are checked in. Now you are ready to start the deliver operation.

Phase 2: Starting to Deliver

Next, you start the deliver operation. The deliver operation is a multistep process where you start to deliver, and re-test the work you delivered with the latest version of the work of your team members. This testing can uncover conflicts that you need to resolve. When all is in order, you then complete the deliver operation.

When you start the deliver operation, you choose to deliver your work by specifying one or more activities or baselines to be delivered. You also select to which target stream to deliver: the default stream or an alternate stream.

If you choose to deliver work that is associated with activities, the deliver operation does the following:

1 Identifies activities that are candidates for delivery.

2 Checks that these activities have been modified since the last deliver operation from your development stream.

3 Checks that the versions being delivered are checked in to the development stream.

If you choose to deliver a baseline, the deliver operation identifies the baselines to be used by the deliver operation.

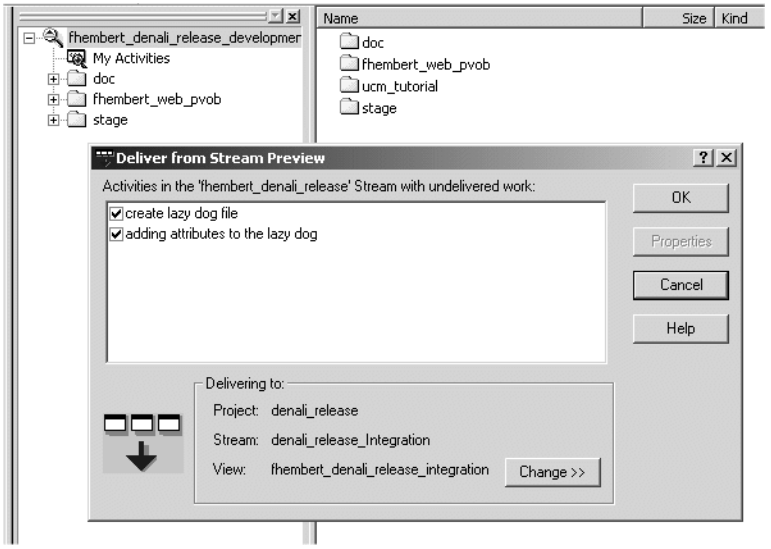
You start the deliver operation from the Project Explorer or the CLI.

Using the Project Explorer, one way to start the deliver operation is the following:

1 Right-click the root directory of your development view.

2 From the shortcut menu, select either **Deliver from stream to default** or **Deliver from stream to alternate target**.

The following example shows the Deliver from Stream Preview dialog box listing two activities that are still to be delivered.

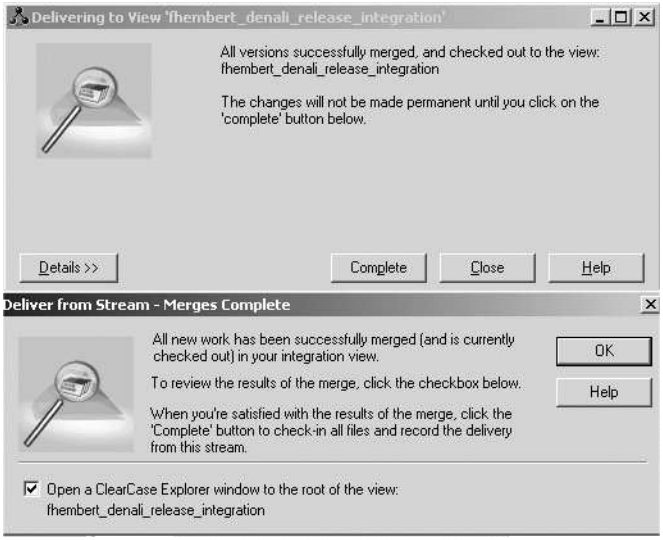


ClearCase found that the two activities created in previous exercises were not delivered.

All other information, namely the project, the stream, and the view where the activities will be delivered, is correct.

3 Click **OK**.

ClearCase starts the deliver operation and displays progress information.



This point in the deliver process does not indicate the deliver operation is complete. You have reached a point where you can verify if the newly created version of the files will enable you to perform a successful test build before completing the deliver operation. If not, you can now make the necessary corrections.

Phase 3: Merging Differences

After you have determined what you will deliver to the integration stream, the deliver operation moves into the merge phase.

In this phase, the deliver operation compares the versions being delivered from the development stream with their counterparts in the target integration stream and invokes the ClearCase Merge Manager as needed.

The ClearCase Merge Manager automatically resolves trivial differences. Trivial differences are those that can be resolved without your manual intervention.

For nontrivial differences, ClearCase Merge Manager prompts you to resolve them manually when the merging occurs.

For learning about the merge operation and working in the Merge Manager, go to the *Merging Your Work* module.

Phase 4: Testing Your Work

Before completing the deliver operation, you must first test the accuracy of your work against the work of your team members.

In the testing phase, you ensure that the work you are about to deliver will not cause any build failures.

If you encounter build issues, you must first resolve them. However, to do so does not require you to cancel the deliver operation that you started. You may need to check out more files in your integration view to resolve build conflicts. However, you don't want to check them back in until you complete the deliver operation.

Phase 5: Completing the Delivery

After you have made sure that the work you deliver is error-free, you are then ready for the final phase, namely to complete the deliver operation.

During this phase, the deliver operation verifies the merges, checks in the changes in the target integration stream, and performs general housekeeping tasks.

Merging Your Work 19

Audience Developers

Platforms Windows, UNIX

Process UCM, base ClearCase

Products ClearCase, ClearCase LT

Merging is the action of combining the contents of two or more files or directories into a single new file or directory.

In UCM, the merging of your work from your private work area to the target stream takes place during a deliver or rebase operation when ClearCase encounters differences that require your manual intervention.

In base ClearCase, you merge your work from your development branch to the integration branch when you want to make your changes visible to others on your team. You merge from the integration branch to your development branch when you want to see other developers' changes.

This module covers the following topics:

- Starting the Merge Manager
- Finding Files to Be Merged
- How ClearCase Merges
- About Trivial and Nontrivial Merges

The information provided in this module is applicable to Windows and UNIX.

Starting the Merge Manager

The Merge Manager is a graphical tool that manages the process of merging one or more ClearCase elements. It automates the processes of gathering information for a merge, starting a merge, and tracking a merge.

In UCM, the merge operation is an integral part of the deliver operation. The deliver operation merges the work in your private development work area with the work of other team members in the target stream. Merge operations can also occur when you rebase your private work area.

Starting the Merge Manager on Windows

From the ClearCase Explorer:

1. Right-click an element in the Files pane and select **Merge Manager** from the shortcut menu.
2. In the Shortcut pane, click **Toolbox > Base ClearCase > Merge Manager**.

Starting the Merge Manager on UNIX

Type the following command at a system prompt:

clearmrgman &

When the Merge Manager starts, it opens a Welcome dialog in which you choose how to proceed.

Finding Files to Be Merged

From the Merge Manager, you can find the files that are to be merged by using the Find Manager.

The Find Manager is a wizard that guides you through the process of finding the files to be merged.

Starting a New Merge Session on Windows

Click the **New** button in the Welcome dialog box.

Alternatively, from the Merge Manager, click **File > New Merge > Find elements to merge**.

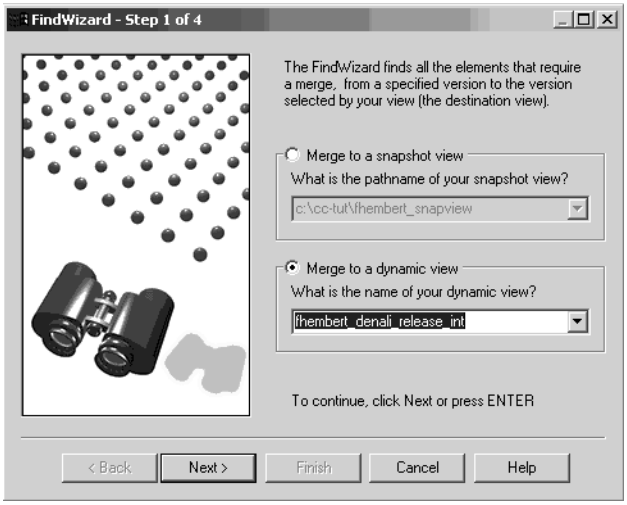
Starting a New Merge Session on UNIX

Click the **New** button in the Welcome dialog box.

Alternatively, from the Merge Manager, click **Merge > Find Elements to Merge**.

Completing the Find Manager Wizard

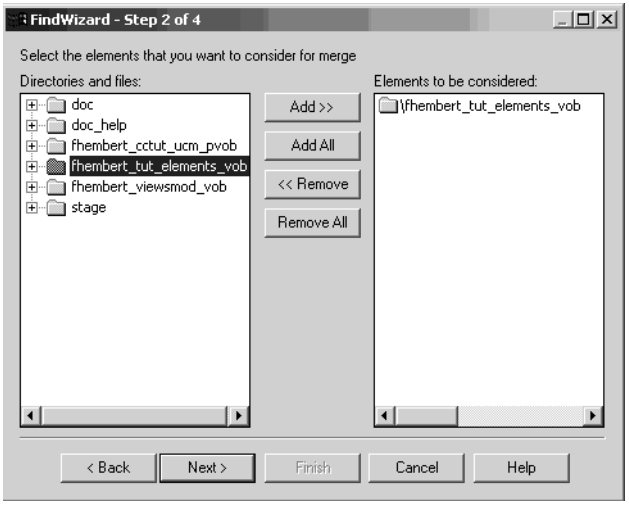
Step 1 of the Find Wizard prompts you to select the destination view.



1 Select the destination view

2 Click **Next**.

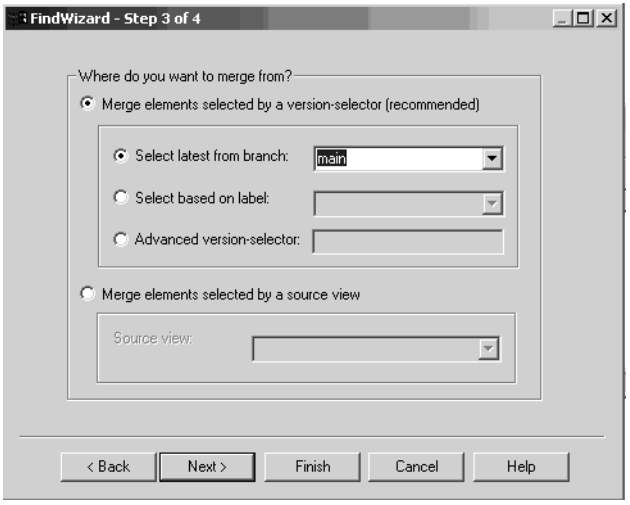
Step 2 of the wizard prompts you to identify the VOB containing the elements to be considered for merging.



1 Select the VOB containing the elements and click **Add.**

2 Click **Next**.

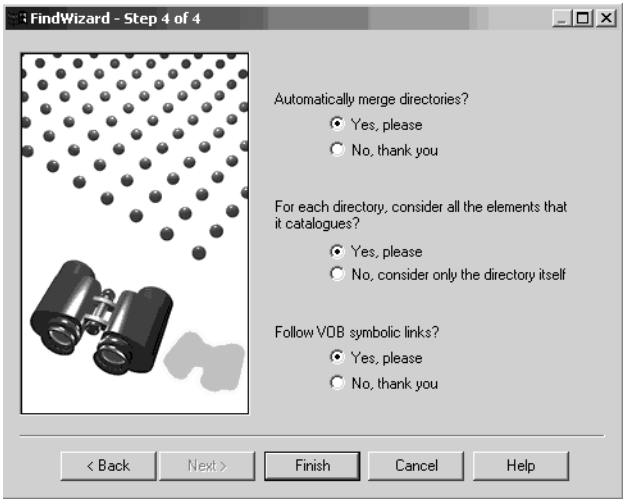
In step 3 of the wizard you determine the merge source: version-selector or source view.



1 Select the source of the elements to be considered for merging.

2 Click **Next**.

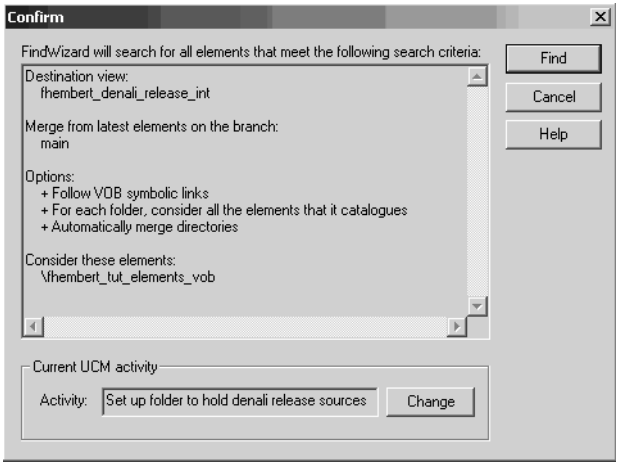
In step 4 of the wizard you specify if the directories should be merged automatically; if the merge should consider all elements; and if the merge should follow symbolic links.



1 Make your selections

2 Click **Finish**.

ClearCase displays a Confirm dialog box.



You can change the UCM activity by clicking the **Change** button.

If your selections are correct, click **Find**.

The files that are candidates to be merged are listed in the Merge Manager.

How ClearCase Merges

To merge versions, ClearCase takes the following steps:

1. It identifies the base contributor.
2. Next, it compares each contributor against the base contributor.
3. For any line that is unchanged between the base contributor and any other contributor, ClearCase copies the line to the merge output file.
4. For any line that has changed between the base contributor and another contributor, ClearCase performs a trivial merge by accepting the change in the contributor. Note, however, that depending on how you started the merge operation, ClearCase may copy the change to the merge output file. However, you can disable the automated merge capability for any given merge operation. If you disable this capability, you must approve each change to the merge output file.
5. For any line that has changed between the base contributor and more than one other contributor, ClearCase requires that you resolve the conflicting difference.

About Merging in UCM

During a deliver or rebase operation, the ClearCase merge algorithm involves the following versions:

- The source contributors: one version from your development stream and one version from the target stream.
- The base contributor: the common ancestor of the source versions.
- The destination version: the merge output, which in a deliver operation becomes a new version in the target stream and in a rebase operation becomes a new version in the development stream.

About Merging in Base ClearCase

The ClearCase merge algorithm uses the following files during a merge:

- Contributors, which are typically one version from each branch you are merging. (You can merge up to 15 contributors.) You specify which versions are contributors.
- The base contributor, which is typically the closest common ancestor of the contributors. (For selective merges, subtractive merges, and merges in an environment with complex branch structures, the base contributor may not be the closest common ancestor.) ClearCase determines which contributor is the base

contributor. The target contributor, which is typically the latest version on the branch that will contain the results of the merge. You determine which version is the target contributor.

- The merge output file or directory, which contains the results of the merge and is usually checked in as a successor to the target contributor. By default, the merge output file is the checked-out version of the target contributor, but you can choose a different file to contain the merge output.

About Trivial and Nontrivial Merges

ClearCase performs two types of merges: trivial and nontrivial merges.

- **Trivial merges**. In a trivial merge operation, the base contributor and the contributor to which you are merging are the same. In the comparison and merge tools, the base contributor is the element against which all other contributors are compared when reporting differences. ClearCase resolves any trivial merges without your intervention.
- **Nontrivial merges**. Differences between the base contributor and contributors are such that your manual intervention is required to resolve the differences. When encountering such differences, ClearCase will prompt for your resolution.

Building Software 20

Audience	Developers
Platforms	Windows, UNIX
Process	Base ClearCase
Products	ClearCase, ClearCase LT

As a developer you are familiar with the make tool or command to build software. Rational ClearCase provides two **make** tools that supplement the features of **make**.

This module covers the following topics:

- About clearmake
- About omake

About clearmake

clearmake is the ClearCase variant of the UNIX **make** utility. It includes most of the features of UNIX System V **make**. It also features compatibility modes, which enable you to use **clearmake** with makefiles that were constructed for use with other popular make variants, including **Gnu make**.

clearmake features the following ClearCase extensions:

- **Configuration lookup**. Configuration lookup is a build-avoidance scheme that is more sophisticated than the standard scheme, which uses time stamps of build objects.

  Configuration lookup also includes automatic detection. This guarantees, for example, correct build behavior as C-language header files change, even if the header files are not listed as dependencies in the makefile.

- **Derived object sharing**. Developers working in different views can share the files created by **clearmake** builds.

- **Creation of configuration records**. Software bill-of-materials records that fully document a build and support the ability to rebuild.

Try It!

1. From the Help, access the Command Reference, and review the **clearmake** page.
2. From the Help, access the Building Software manual from the Online Manuals section and review one or more chapters of your choice.

About omake

omake is another ClearCase utility to build, maintain, update, and regenerate groups of programs. It includes many of the configuration management facilities provided by the **clearmake** utility.

omake also features emulation modes, which enable you to use **omake** with makefiles that were constructed for use with other popular make variants, including Microsoft NMAKE, Borland Make and the PVCS Configuration Builder (Polymake.)

omake is intended for use in dynamic views. You can use **omake** in a snapshot view, but none of the features that distinguish **omake** from ordinary make programs, such as build avoidance, build auditing, derived object sharing, and so on, works in snapshot views.

Try It!

1. From the Help, access the Command Reference, and review the **omake** page.
2. From the Help, access the *OMAKE Guide* from the Online Manuals section and review one or more chapters of your choice.

Rebasing Your Private Work Area

21

Audience	Developers
Platforms	Windows, UNIX
Process	UCM
Products	ClearCase, ClearCase LT

Project managers organize delivered activities into baselines. When baselines reach a satisfactory level of stability, after several cycles of testing and fixing defects, project managers will designate a recommended baseline.

As a developer, you need to update your development work area with the recommended baseline as soon as it becomes available.

This module covers the following topics:

- Overview of the Rebase Process
- Performing a Rebase Operation
- Undoing a Rebase Operation (Windows and UNIX)

Overview of the Rebase Process

The rebase process involves the following tasks:

1. **Prepare your work area**. You need to prepare your work area before performing a rebase operation. This involves finding all your checkouts and checking them in because the rebase operation cannot start from a view that contains checkouts. You can also check for differences between the earlier checked-in version, and see the check-in comments.
2. **Start the rebase operation**. ClearCase will present you with the latest recommended baseline.
3. **Be prepared to perform merge operations**. If another team member has modified and delivered a version of the same element you modified in your development

stream, you will have to perform a merge operation when you rebase to a baseline that contains the delivered version. ClearCase merges all nonconflicting differences and will prompt you to decide how to resolve conflicting differences.

4 **Test your undelivered work after the rebase**. After ClearCase reconfigures your work area and any conflicting merges have been resolved, you must verify that any undelivered work builds successfully in your work area using the most up-to-date elements of the new baseline. In the event of build errors, you will first need to resolve any conflicts before you can proceed to delivery your work.

5 **Complete the rebase operation**. Only when you are satisfied with your test builds will you complete the rebase operation.

Performing a Rebase Operation

The following sections cover the following tasks:

- Preparing Your Development View
- Starting the Rebase Operation
- Rebasing Your Development Work Area
- Merging Versions
- Testing
- Completing the Rebase Operation

Preparing Your Development View

Listing Checkouts

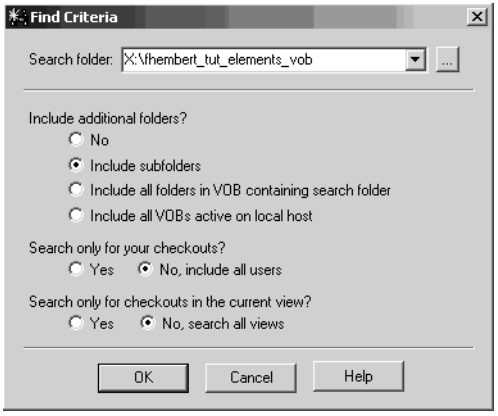
This task involves searching for files that are still checked out so that you can check them in.

Listing Checkouts on Windows

To search for checkouts from the ClearCase Explorer:

1 In the Folders pane, select the top level folder that contains the checkouts.

2 Right-click and select **Find Checkouts** to open the Find Criteria dialog box.



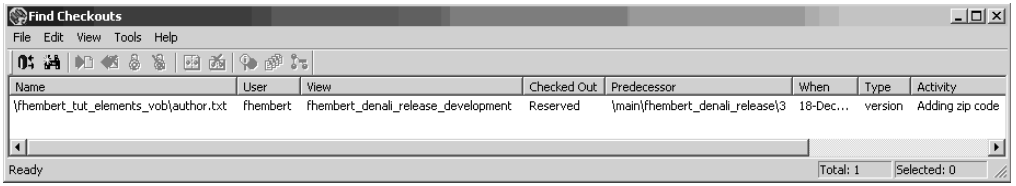
The Search folder lists the folder you selected in Step 1.

The Include options let you narrow or widen the scope of the search. By default, the option to include subfolders is selected.

The search option lets you widen the search to include checkouts from other users (select **No, include other users**) and all views (select **No, include all view**).

3 Click **OK** to view the search results in the Find Checkouts dialog box.

If any checkouts are found, the Find Checkouts dialog box identifies the following (from left to right):



- The name of the file and its full path
- The username of the individual who checked out the files
- The view tag from which the files were checked out
- If the file was checked out reserved or unreserved
- The the name of the branch and version number of the files' predecessors
- Work or activity performed
- If in a MultiSite environment, the Master Replica of Branch (not visible on the illustration)
- The checkout comments, if they were entered at checkout time (not visible on the illustration)

Listing Checkouts on UNIX

The **lscheckout** command lists any outstanding checkouts. The command has more parameters than shown in this section, but the command as shown corresponds to the options selected using Find Checkouts from the ClearCase Explorer.

At a system prompt, type this command:

cleartool lscheckout -cview -me -avobs

-cview restricts the list to checkouts made in the current view.

-me restricts the listing to your checkouts only.

-avobs includes checkouts in all mounted VOBs

System response is similar to the following:

\jdoe_denali\mck\rel1\xyz.c, where

- **\jdoe_denali** is the current view
- **\mck** and **\rel1** are directories under source control
- xyz.c is the name of the checked-out file

Listing Content Differences

After your find outstanding checkouts, you need to investigate the differences. To do so, you use Compare with previous version option. This option is available from the **Tools** menu in the Find Checkouts dialog box or from the shortcut menu.

Note: When you attempt to compare nontext files such as binary and graphic files, an error message informs you can only use the compare option on text files.

Listing Content Differences on Windows

To compare with the previous version from the Find Checkouts dialog box:

1. Select a file in the Find Checkouts dialog box
2. Right-click and select **Compare with previous version** to start the Diff Merge tool.

   The Diff Merge tool lists the current and previous versions of the selected file.

Listing Content Differences on UNIX

To display differences between the checked-out version and its predecessor, use the **diff** command:

cleartool diff -predecessor *filename*

For example, **cleartool diff -predecessor** xyz.c

Listing Checkin Comments

Listing Checkin Comments on Windows

The Find Checkouts dialog box lists the checkin comments in the **Comments** field.

Listing Checkin Comments on UNIX

To display the checkin comments, use the **lshistory** command:

cleartool lshistory *filename*

For example, **cleartool lshistory** xyz.c

Checking Files In

Any checked-out files must be checked in, or the checkout canceled.

Checking Files In on Windows

You use **Tools > Check In** to check in a file or **Tools > Cancel checkout** to cancel a checkout in the Find Checkouts dialog box, or from the shortcut menu.

To check in the file from the Find Checkouts dialog box:

1. Select a file.
2. Right-click and select either **Check In** or **Cancel checkout** from the shortcut menu.
3. When checking in, add or replace the checkin command and click **OK**. When canceling a checkout, deselect the keep option if you do not want to keep a private copy of the file, and then click **OK**.

Checking Files In on UNIX

The **cleartool checkin** command checks in a file. The command format is:

cleartool checkin *filename*

The **cleartool uncheckout** command cancels the checkout. The command format is:

cleartool uncheckout *filename*

Starting the Rebase Operation

After preparing your development stream as outlined in the preceding section, you can start the rebase operation.

The rebase operation involves the following tasks:

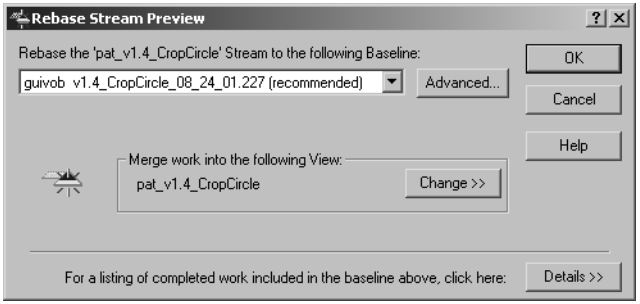
- Selecting the recommended baseline.
- Merge operations; this may require to stop and resume the rebase operation to enable you to perform test builds in your work area.

The following guides you through a rebase operation.

Starting a Rebase Operation on Windows

In the ClearCase Explorer, right-click the root directory of the development view and click **Rebase Stream** to open the **Rebase Stream Preview** dialog box (Figure 5).

Figure 5 The Rebase Preview Dialog Box (Windows)



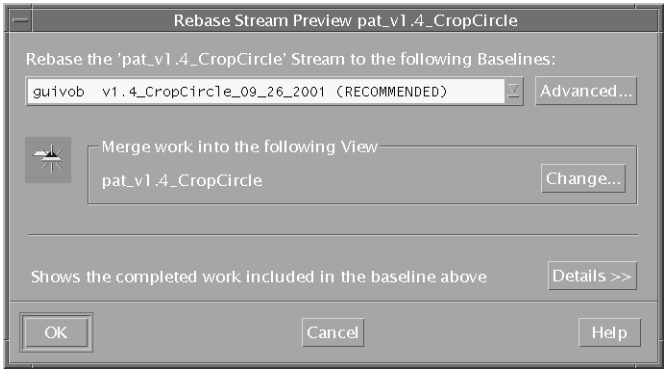
Starting a Rebase Operation on UNIX

At a system prompt, type:

cleartool rebase -graphical

See Figure 6.

Figure 6 The Rebase Preview Dialog Box (UNIX)



By default, the dialog box presents the recommended baseline.

It is important to verify that the dialog box displays your development view under **Merge work into the following view**. If you attached multiple views to your development stream, use the **Change** button to choose one view for the rebase operation.

Clicking the **Details** button will show the list of activities in the baseline.

4 Click **OK**.

Rebasing Your Development Work Area

In rebasing your work area to the recommended baseline, ClearCase changes your development stream configuration to select the new recommended baseline. The stream continues to select any undelivered activities created in the development stream.

For snapshot development views, ClearCase starts an update operation to copy the versions of the new baseline into your view. If other snapshot views are attached to your development stream, you must update these also so that they contain the changes in the stream.

For dynamic development views, ClearCase shows the versions of the new baseline without requiring an update operation.

Merging Versions

If another team member modified and delivered a version of the same file, you must perform a merge operation when you rebase to a baseline that contains the delivered version.

As it does in a deliver operation, ClearCase merges all nonconflicting differences.

For conflicting differences, ClearCase will prompt you to start the Merge Manager to resolve the conflicts. For information on merging, go to the *Merging Your Work* module.

Testing

After all merge conflicts are resolved, you need to verify that any undelivered work in your work area builds correctly with the element versions resulting from the new baseline.

Testing your undelivered work will undoubtedly involve checking files in and out, and performing several builds until no build errors are reported.

Completing the Rebase Operation

Completing a rebase operation consists of two tasks:

- Checking in any merge results, and
- Completing the rebase operation. This is accomplished by clicking the **Complete** button in the Merge Manager (Windows) or the clearmrgman (UNIX) windows.

Undoing a Rebase Operation (Windows and UNIX)

As long as you have not completed a rebase operation, you can undo it.

To undo a rebase operation, type this command at a Windows or UNIX system prompt:

cleartool rebase -cancel

Checkouts performed by the rebase operation in your development view will be canceled.

Index

A

activities
creating on UNIX 77

B

base ClearCase
about 17
comparing with UCM 15
merging files in 164
baseline comparison 142
baseline types 142
baselines
comparing on UNIX 143
comparing on Windows 142
create on Windows 143
creating 143
defining an integration baseline 81
definition 142
promoting 145
promotion levels 145
recommending 146
working with 141
branches
config spec rules 148
creating 150
definition 147
instantiating 150
use of config spec in 148
using config spec in subbranches 149
when to use 148
working with 147
build commands 167–168
building software 167

C

checkin requirements 127
checkout history 137–138
checkout requirements 127
checkouts
canceling 136
canceling on UNIX 137
canceling on Windows 136
file history 138
finding 133
history of 137
reserved and unreserved 130
searching for, on UNIX 135
searching for, on Windows 133
ClearCase commands 27
ClearCase reference pages 29
clearmake command 167
cleartool apropos 28
command line interface 27
command modes 27
interactive mode 28
single-line mode 27
commands
clearmake 167
cleartool apropos 28
command line interface 27
omake 168
components
creating 50

creating on UNIX 56
creating on Windows 54
creating to store elements 53
select the project's 93
selecting the project's 90
config spec 121, 148
changing 125
changing on UNIX 125
changing on Windows 125
default rules in 148
how it works 121
include files 124, 149
the role of 123
understanding the default 124
using in subbranches 149

D

delivering
completing a deliver operation 157
merging differences 157
performing a deliver operation 154
preparing for a deliver operation 154
process overview 153
starting a deliver operation 155
testing your work 157
Delivering your work 153
development stream
creating 91, 94
creating in UCM 86
directory structure
creating 78
creating on UNIX 79
creating on Windows 78
documentation resources 11

F

files
about hijacking 139
adding to source control on UNIX 129
adding to source control on Windows 128
cancel ing checkouts on Windows 136
canceling checkouts 136
canceling checkouts on UNIX 137
checkin and checkout of 127
checkin and checkout requirements 127
checking in 140
checking in during rebase 173
checking in on UNIX 140
checking in on Windows 140
checking out on UNIX 132
checking out on Windows 131
checkout history on UNIX 138
checkout history on Windows 138
finding checkouts 133
finding for merging 160
history of checkouts 137
merging during a deliver operation 157
searching for checkouts on UNIX 135
searching for checkouts on Windows 133
when to check in files 139

H

Hijacked files 139

I

include files
for config spec 149

role of 124

M

merging
about nontrivial merges 165
about trivial merges 165
during a deliver operation 157
finding files for 160
how it works 164
in base ClearCase 164
in UCM 164
starting on UNIX 161
starting on Windows 160
starting the Merge Manager 159
merging your work 159

O

omake command 168

P

project component
creating 52

R

Rational University 13
rebasing
checking in files 173
listing checkin comments 173
listing checkouts for rebase 170
listing checkouts on UNIX 172
listing checkouts on Windows 170
listing content differences on UNIX 172
listing content differences on Windows 172
merging versions during rebase 175
performing a rebase operation 170
preparing for 170
process overview 169
starting on UNIX 174
starting on Windows 174
testing after 175
undo 176
your development work area 175
your private work area 169
reference pages
online access 29
searching in 31
roadmap 2
for developers 5
for project managers 3

S

snapshot view
updating 117
updating on UNIX 118
updating on Windows 118
source control
adding to 78
starting the tutorial 9

T

tutorial
access to PDF format 12

exercise requirements 8
how to start 1
location of files 9
printing 10
roadmap 2
software requirements 7

U

UCM
comparing with base ClearCase 15
creating a development stream 86
creating a PVOB 44
creating a PVOB on UNIX 49
creating a PVOB on Windows 44
creating and setting activities 77
developer tasks 83
integrating with ClearQuest 71
joining a project 83–84
merging files in 164
process overview 16
project set-up 43
workflow 15
UCM project
creating 58
creating on UNIX 66
creating on Windows 59
joining on UNIX 93
joining on Windows 84

V

view
specifying the location of your integration view 90
view directories
about 103
view types 104
views
about 103
about development views 103
about integration views 104
considerations for dynamic views 105
considerations for snapshot views 105
controlling the contents of 121
creating 107
for your development stream 88
for your integration stream 88
creating a development stream 95
creating an integration stream 95
creating an integration view 72
creating an integration view on UNIX 76
creating an integration view on Windows 74
creating on UNIX 114
creating on Windows 107
list active 118
list active, on UNIX 119
list active, on Windows 119
names for 104
remove 119
remove on UNIX 120
remove on Windows 119
snapshot views
updating on UNIX 118
specifying the location of your development view 89, 92
specifying the location of your integration view 92
starting on UNIX 116
starting on Windows 115
stopping on UNIX 116
stopping on Windows 116

updating snapshot views 117
on Windows 118
which to create 105
working with 103
VOBs
creating 33
creating on UNIX 41
creating on Windows 34
mounting 99
mounting on UNIX 100
mounting on Windows 99
types 33
unmounting 99–100
unmounting on UNIX 101
unmounting on Windows 100
VOB Creation Wizard 34