Working in Base ClearCase

Contents

Working in Base ClearCase

Figures

Rational ClearCase Concepts 1

Rational ClearCase provides a flexible set of tools that your organization uses to implement its development and change management policies. To use these tools, you need to understand the following concepts:

- ClearCase elements
- Views, versions, and VOBs
- Parallel development

If your project uses the base ClearCase-ClearQuest integration, you need to understand the following concepts:

- Activities
- ClearQuest user databases
- Associating versions and activities

Recommended Reading Paths

Read this chapter first. Then, if you want to start working immediately, use Help to learn as you go. Or, if you prefer a more structured approach, use the remainder of *Working in Base ClearCase* as a guide through the development cycle of your organization. The sections titled *Under the Hood* provide detailed information and suggest ways to become an advanced ClearCase user.

To Access Online Information

To start ClearCase Help, click **Start > Programs > Rational Software > Rational ClearCase > Online Help**.

The help provides links to introductory material. To access the tutorials, click **Start > Programs > Rational Software > Rational ClearCase > Tutorial**.

For more information, see the discussion of online documentation in the Preface.

ClearCase Elements and Activities

Items under ClearCase source control (version control) are generally referred to as elements. An element can be a design model, C++ source file, Visual Studio project, or a DLL. Elements are typically the objects on which you do work.

If your project uses the base ClearCase-ClearQuest integration, the things that you do in performing your work and their status are captured in *activities*. Activities are stored in ClearQuest user databases. Activities represent work assigned to you, and allow your project manager to gauge your progress.

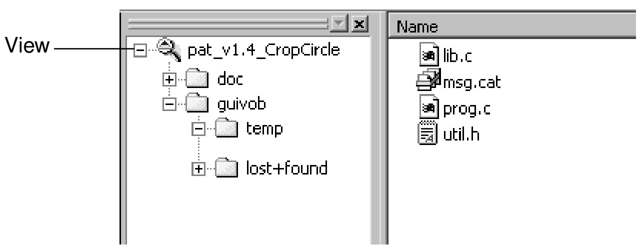
This section describes the following concepts in understanding elements and activities:

- Views
- Elements and versions
- ClearQuest user databases

ClearCase Views

To access files under ClearCase control, you set up and work in a *view*. Figure 1 shows a view named **pat_v1.4_cropcircle** as seen from ClearCase Explorer.

Figure 1 A View as Seen from ClearCase Explorer



A *view* shows a directory tree of specific *versions* of source files. The view **pat_v1.4_cropcircle** is a directory tree of source files for developing release 1.4 of the Cropcircle software product.

Snapshot Views and Dynamic Views

ClearCase includes two kinds of views:

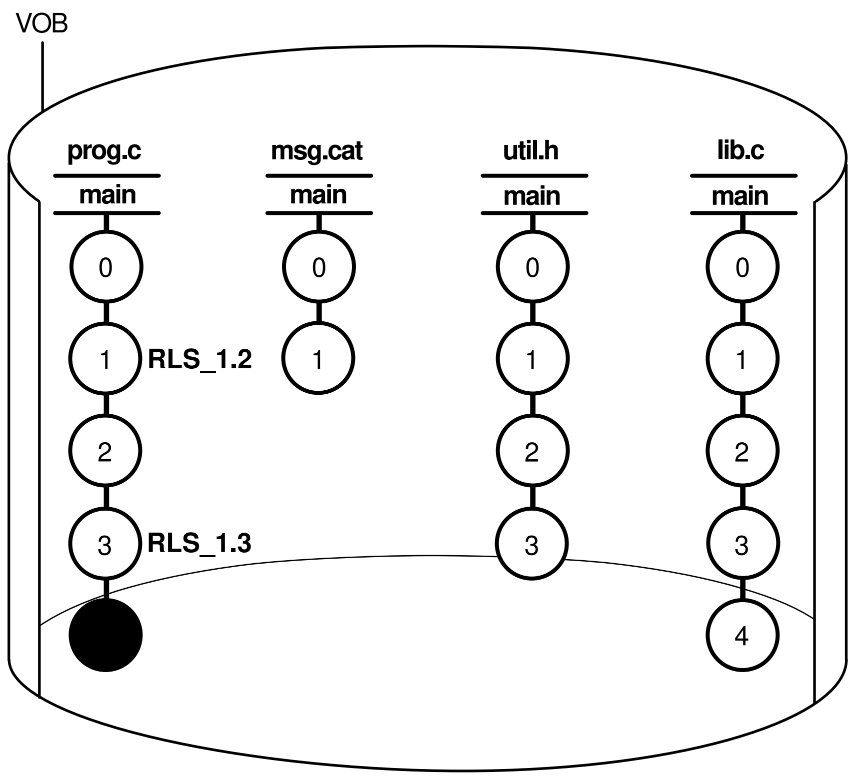
- *Snapshot views*, which copy files from data repositories called *VOBs* (*versioned object bases*) to your computer.
- *Dynamic views*, which use the *multiversion file system* (MVFS) of ClearCase to provide immediate, transparent access to the data in VOBs. (Dynamic views may

not be available on all platforms. For more information, see the Help. For information about starting Help, see *To Access Online Information* on page 1.)

Versions, Elements, and VOBs

Each time you revise and check in a file or directory from a view, ClearCase creates a new *version* of it. Files and directories under ClearCase control (and all of their constituent versions) are called *elements* and are stored in *VOBs*. Figure 2 illustrates a VOB that contains the file elements prog.c, util.h, msg.cat, and lib.c.

Figure 2 A VOB Contains All Versions of an Element



Depending on the size and complexity of your software development environment, ClearCase *elements* may be distributed across more than one VOB. For example, the elements used by the documentation group are stored in one VOB, while the elements contributing to software builds are stored in a different VOB.

Selecting Elements and Versions

A set of rules called a configuration specification, or *config spec*, determines which files are in a view.

Config Specs for Snapshot Views

Config specs for snapshot views contain two kinds of rules: *load rules* and *version-selection rules*. Figure 3 illustrates how the rules in a config spec determine which files are in a view.

Config Specs for Dynamic Views

Dynamic views use version-selection rules only (and ignore any load rules). A dynamic view selects all elements in all VOBs activated on your computer, and then uses the version-selection rules to select a single version of each element. Instead of copying the version to your computer as a standard file, the view uses the *MVFS* (multiversion file system) to arrange the data selected in the VOB into a directory tree.

Criteria for Selecting Versions

The rules in the config spec constitute a powerful and flexible language for determining which versions are in your view. For example, version-selection rules can specify the following criteria:

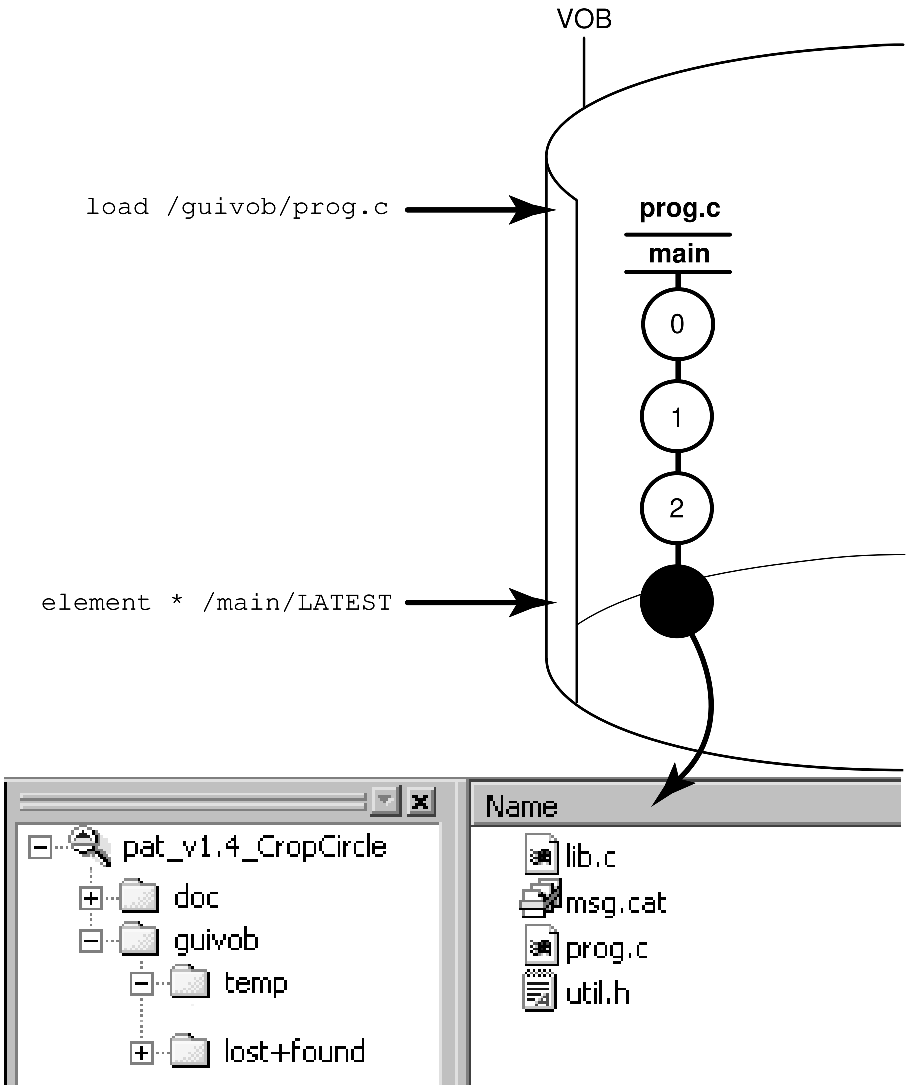
- The latest version.
- A version identified by a *label*.

  A label is a text annotation that you can attach to a specific version of an element. Usually, your project manager attaches a label to a set of versions that contributed to a specific build. A typical config spec rule uses version labels to select versions:

  ```
  element * BASELINE_1
  ```

  For example, if your project manager attaches version label **BASELINE_1** to a version of element prog.c, any view configured with this rule selects the labeled version (unless some rule earlier in the config spec matches another version of prog.c). For more information about labels, see *Managing Software Projects*.

Figure 3 Config Spec Selecting Versions



- A version identified by a *time rule*, that is, a version created before or after a specific time.

The version-selection rules are prioritized. For example, the view can try to select a version identified by a label first, and if no such version exists, the view can select a version based on a time rule.

Version Labels in Version-Extended Paths

In addition to affecting the way the element appears in views, labeling a version of an element also provides a way to access the version with a version-extended path (see *The Version-Extended Path* on page 70). Labeling a version effectively adds a new hard link to the version in the extended namespace. If you attach version label **R4.1A** to version **/main/rls4/12** of element lib.c, these paths are equivalent:

```
lib.c@@/main/rls4/12
lib.c@@/main/rls4/R4.1A
```

In addition, a third path is *usually* equivalent:

```
lib.c@@/R4.1A
```

This version-extended path is valid if it is unambiguous, that is, if no other version of lib.c is currently labeled **R4.1A**. (This is usually the case because, by default, label types are restricted to being used once per element. See the description of the **–pbranch** option in the **mklbtype** reference page in the *Command Reference*.)

Learning the Config Spec Syntax

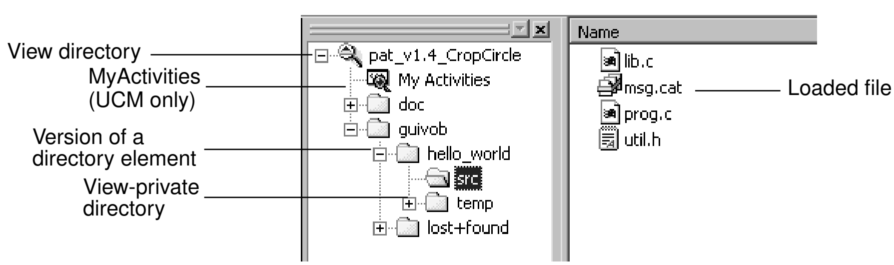
Usually only one or two members of your software team learn the syntax for these rules and create config specs for everyone on the project to use. For more information, see *Managing Software Projects* and the **config_spec** reference page in the *Command Reference*.

View-Private Objects

In addition to versions of source files, a view also contains file-system objects that are not under ClearCase source control, such as temporary files that you create while developing your source files and other objects. These non-ClearCase file system objects are called *view-private files* and *view-private directories*.

Figure 4 shows the **pat_v1.4_cropcircle** view with the objects labeled.

Figure 4 A View Contains Versions of Elements and View-Private Objects



The Base ClearCase-ClearQuest Integration

Your ClearCase administrator can integrate base ClearCase projects with a ClearQuest user database. You then associate versions of ClearCase elements on which you are working and change requests in a ClearQuest user database.

The Base ClearCase Schema and ClearQuest User Databases

ClearQuest stores data in schema repositories and user databases. A schema defines the types of records in the user database and other attributes of the user database. ClearQuest stores all schemas in a schema repository. The ClearQuest user database stores your change-request data.

The base ClearCase schema package provides additional information in your ClearQuest records to track associations with ClearCase versioned objects. To support associations between base ClearCase versions and ClearQuest record types, the schema repository needs to have this schema package applied to selected record types. Then the target ClearQuest user database must be updated to add the new fields provided by the package.

Your ClearQuest user database may include different record types for different purposes. The record type used by the SAMPL ClearQuest user database supplied with the base ClearCase-ClearQuest integration is called a defect, but with the base ClearCase schema package installed, any change-request record type can be used.

ClearCase Triggers and Change Requests

The base ClearCase-ClearQuest integration consists of ClearCase triggers that fire when you check out a file, cancel a checkout, or check in a file. Your ClearCase administrator installs the integration triggers into each target VOB. The integration associates one or more change requests with one or more versions stored in one of the target VOBs.

- A single change request may be associated with more than one version. The set of versions that implement the requested change is called the *change set* for that request.
- A single version may be associated with more than one change request. These change requests are called the *request set* for that version.

Uses of the Base ClearCase-ClearQuest Integration

The base ClearCase-ClearQuest integration provides either of the following:

- A window interface for users of the graphic user interfaces (GUIs) of ClearCase, such as the Explorer of ClearCase or Windows (on Windows computers)
- A text-based user interface for users of the **cleartool** command-line interface

The base ClearCase-ClearQuest integration has triggers on checkin, checkout, and cancel checkout operations. Working on versions, you can do the following:

- Associate a version with one or more change requests when you check out or check in files.
- List the request sets that are associated with a project over a period of time, list the change requests associated with a specific version, and see the related hyperlinks.

Using the ClearQuest user database, you can do the following:

- View the change set for a change request.
- See the files that fix a specific problem.

ClearCase administrators can do the following:

- Install the related triggers in a VOB and set a policy for each VOB.
- Specify that you are prompted on checking out a version, checking in a version, or both.
- Specify that prompting occurs only for some VOBs, branch types, or file types. Associations of checked-in versions with change requests can be either optional or required.

A ClearCase administrator adds the base ClearCase schema package to a schema and applies the change to one or more change-request record types. The policy that the administrator sets for one or more VOBs specifies the conditions under which you are prompted to associate versions with change requests.

Parallel Development

The combination of config spec rules, views, VOBs, and *branches* (described in Chapter 5, *Working On a Team*) provide the basis for *parallel development*, a strategy in which an organization can work on multiple versions of the same source file concurrently. For example, you are working on release 1.4 of a software product, and you want to experiment with the graphic user interface as a result of feedback from usability testing. You can create a view that isolates your modifications from the rest of the release 1.4 development project. Although you work with the same set of files used in the official builds, the versions of the files that you create from this view evolve separately from the versions used in the official builds. When you are satisfied with your usability modifications, you can use ClearCase tools to merge your work with the files used in the official release 1.4 build.

About Rational ClearCase Explorer

Rational ClearCase Explorer is a developer's tool that supplies a coherent, adaptable, and customizable interface to your software development environment, providing:

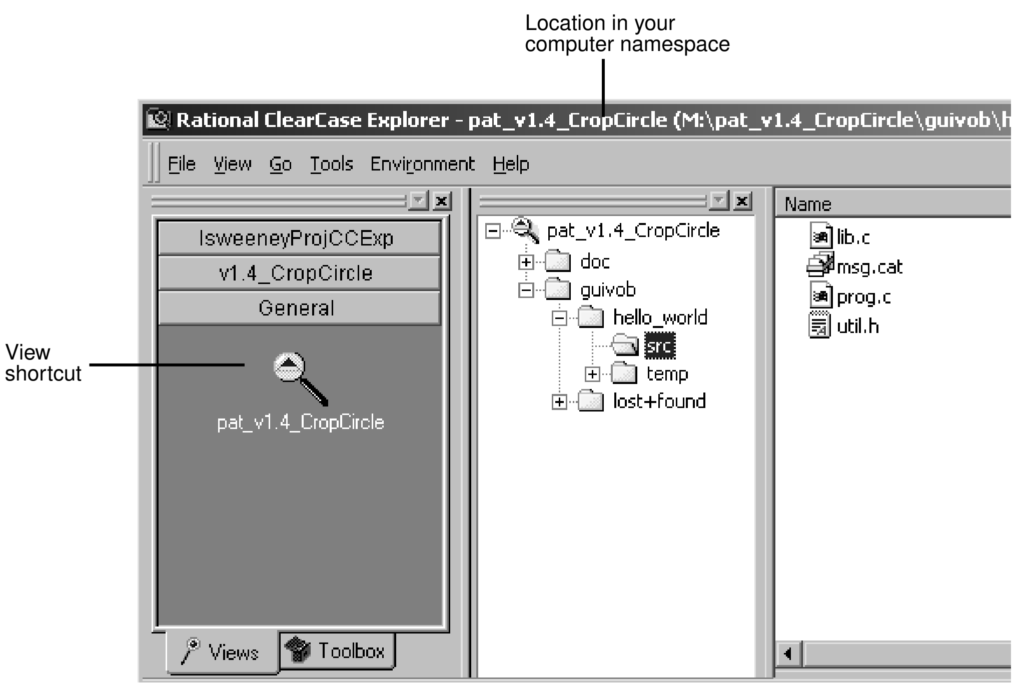
- Organization of ClearCase developer tools
- Access to the ClearCase environment
- Flexible navigation mechanisms

The ClearCase Explorer interface is similar to that of Windows Explorer and has dockable panes. ClearCase Explorer supports shortcuts to ClearCase tools, to ClearCase documentation, and to ClearCase objects.

The ClearCase Explorer Window

In its initial state, the ClearCase Explorer window has three panes: Shortcut, Folder, and Details (Figure 5).

Figure 5 A Base ClearCase View From ClearCase Explorer



Shortcut Pane

The leftmost pane, called the Shortcut pane, is dockable and provides tabs to access distinct sections called pages. The **Views** tab accesses icons for shortcuts to ClearCase views and folders in the Folders pane. The **Toolbox** tab accesses icons for shortcuts to ClearCase tools, configurable user tools, and URLs. The icons display ToolTips describing the shortcuts.

Each page organizes icons that provide shortcuts to the items accessible in that page. Click an icon to activate that item. For example, click a view shortcut to start a view. Right-click on the page title to select whether large or small icons are displayed in all pages. If all icons cannot fit on a page, a downward- or upward-facing arrow at the top and bottom of the page provides a scrolling mechanism to access the hidden icons.

Folder Pane

The middle pane, called the Folder pane, is dockable and accesses different tree controls that display the folder and file hierarchy for a given ClearCase view. In the Folder pane, you can navigate to objects in a view that you started.

Details Pane

The rightmost pane, called the Details pane, is not dockable and displays variable content in named tabs having content-specific icons. The content in the Details pane is based on an action in the Folder pane or based on a URL accessed from the **Toolbox** tab.

Information Pane

An optional pane, called the Information pane, is not dockable. When a view is active, the Information pane can provide context-dependent ClearCase state and processing information and access to introductory Help topics.

Toolbars

Similar to the toolbars on your desktop task bar, ClearCase Explorer optionally displays a movable toolbar for elements. The toolbar displays shortcuts to navigation and source control commands and is context sensitive. If you move the toolbar to the desktop, its title bar reads **Standard Toolbar**. You can define your own tailored toolbars.

Status Bar

An optional area in a fixed position at the bottom of the ClearCase Explorer window, the status bar displays information dependent on the current context.

ClearCase Explorer Tool Organization

ClearCase Explorer supports access to tools in several ways:

- Clicking a tab in the Shortcut pane
- Right-clicking in the Folder pane or Details pane
- Clicking an icon in the toolbar (see *Toolbars* on page 10)

Toolbox Tab

The **Toolbox** tab accesses icons for shortcuts to run both ClearCase tools and user-selectable tools and to access documentation and URLs. Tools are organized by tool groups and divided into two pages, **UCM** for project-related tools and **Base ClearCase** for tools not related to UCM projects. The tools provided by default are classified as developer tools. Other ClearCase tools classified as project manager or administrator tools are accessible in the Windows task bar by clicking **Start > Programs > Rational Software > Rational ClearCase > Administration**.

The **Toolbox** tab also presents a **Getting Started** page that provides access to Help, online versions of printed documentation, and tutorials.

Shortcut Menu Support

ClearCase Explorer provides access to tools through the following types of shortcut menus:

- For the Windows Explorer integration for ClearCase in the Folder pane and in the Details pane
- For user customizations of the ClearCase shortcut menu
- For customizing the Shortcut pane

When you right-click in the Folder pane or in the Details pane, a context-sensitive shortcut menu offers access to file handling operations and element processing functions. Provided by the ClearCase-Windows Explorer integration, this shortcut menu contains commands that are activated based on the state of the item you select. If the item is under ClearCase source control, you have available ClearCase commands for element processing functions and Windows commands for file-handling operations. If the item is view-private, you have available file-handling operations and the ClearCase command **Add to Source Control**.

You can use the Context Menu Editor to customize the ClearCase shortcut menu. The editor allows a ClearCase administrator or you to add commands to or modify commands on the ClearCase shortcut menu and add shortcuts to or remove shortcuts from the menu. The steps to do this are described in the Help. You can also add tool shortcuts to your **Send To** folder to access your favorite integrated development environment (IDE), your favorite editor, or a command shell. Any tool shortcuts newly added to your **Send To** folder appear on the shortcut menu **Send To** command the next time ClearCase Explorer starts.

When you right-click in the Shortcut pane, another type of context-sensitive shortcut menu enables you to add and control your own tool groups and tool and URL shortcut icons. If the **Toolbox** tab is active, the shortcut menu commands enable you to add icons for shortcuts to your favorite tools and URLs and add page controls for your own tools. If the **Views** tab is active, with the shortcut menu, you can add and remove view shortcuts and add pages to organize view shortcut icons. And you can perform some ClearCase operations on views. The ways to do this are described in the Help.

Tool Context

A tool started from the Shortcut pane or from a shortcut menu command can have view context, if applicable. For example, starting the History Browser shows the event log for the element currently selected in the Details pane. If you start a project file (for example, a .DSW or a .VPB file) from within ClearCase Explorer, you can perform element processing functions from within the IDE. As you work with ClearCase objects within the IDE, you can refresh the ClearCase Explorer display to see the state change of the elements in the Details pane.

Site Shortcuts

ClearCase Explorer supports defining tool and URL shortcuts that can be shared among ClearCase users. Called site shortcuts, this type of shareable shortcut is defined externally by a project manager or ClearCase administrator and can be made available to you. (View shortcut definitions can only be locally defined.) The external location for the shortcut definitions is specified during ClearCase site preparation. The location for the definition file is stored in the Windows registry on each ClearCase client computer. You take advantage of these site shortcuts by configuring ClearCase Explorer to import the shortcut definitions at startup. At any time, you can reset system tool shortcuts to remove all modifications or deletions you made to any system or site shortcuts.

ClearCase Explorer Navigation

Through the **Views** tab in the Shortcut pane, ClearCase Explorer provides means for accessing and navigating among ClearCase views.

View Access

You access a non-UCM view by clicking the **Views** tab in the Shortcut pane and clicking the **General** page. By default, in the **General** page are icons for shortcuts to non-UCM views that you own.

View Startup

Clicking a view shortcut activates (starts) the view, activates the Details pane, and displays the Folder pane. This establishes a *current* view. ClearCase Explorer displays in the Folder pane a hierarchy rooted at the view tag, and, as subfolders under the view, the contents of the view. For a dynamic view, the subfolders are for all mounted VOBs. For a snapshot view, the subfolders are for all VOBs rooted at the view. By default, in a snapshot view, only loaded elements are visible. You can change this default so that unloaded elements are displayed. The Details pane is activated and lists the subfolders that are exposed in the Folder pane.

The title bar shows the currently selected path within the view. The status bar shows the view tag associated with the current view, the number of items listed in the Details pane, and the number of listed items that are selected.

As you click a subfolder, the subordinate elements display in the Details pane showing ClearCase details about the items, element and version information. In the Details pane, elements possessing unique characteristics (like a file under version control or an eclipsed version) display with visual characteristics familiar in other ClearCase tools.

URL Access

You access a URL by clicking the **Toolbox** tab in the Shortcut pane, the page on which the shortcut is located, and the shortcut to the URL. An Internet Explorer control is activated in the Details pane and the target location is accessed. The Details pane renders the HTML content of the target. The URL tab on the Details pane shows the path to the URL. The Folder pane remains unchanged, but the Information pane, if displayed, is occluded. You can navigate HTML pages clicking **Go > Back** and **Go > Forward**. The status bar maintains the context for the previously selected view.

Default Context

Clicking an icon in the Shortcut pane establishes that shortcut as the default of its type: view. ClearCase Explorer saves this default context when the tool exits, so that it is restored when the tool restarts.

Like the Details pane, the Folder pane maintains navigation context for the current view. Clicking a URL shortcut does not disturb the navigation context. This context is maintained across ClearCase Explorer starts.

History of Visited Locations

As you navigate from one location to another, ClearCase Explorer stores the path for each of your visits. To return to a previously visited location, click **File** and, from the numbered list of paths (ordered from most recently visited to first visited), select the path.

Display of ClearCase Element Information

The tree controls in the Folder pane provide navigation to ClearCase objects in a view (see *ClearCase Explorer Navigation* on page 12).

Displaying ClearCase Element Information

If you activate the Folder pane, the default view is represented as a top-level folder in the Folder pane with the context of the last navigation. Expanding the subfolders provides details about the ClearCase objects accessible in the view. Selecting a folder displays fields of file system and ClearCase information in the Details pane. Use the controls in the Folder pane to expand and close the subfolders and navigate to the element that you want to work on. In the Details pane, elements possessing unique characteristics (like a file under version control or an eclipsed version) display with visual characteristics familiar in other ClearCase tools.

You can control which fields are displayed in the Details pane. Because gathering some data for display takes more processing time and affects tool performance, you may not want to display all possible fields. You can also use standard Windows techniques to sort the data in the columns. Any customizations you make are persistent across invocations of the tool.

In the Folder pane and Details pane, normal Windows Explorer functions such as delete, cut and paste, and drag and drop are supported. Delete (for example, pressing the DELETE key) performs a **cleartool rmname** operation on a selection. Rename does a **cleartool mv** operation. Cut and paste and drag and drop operations enable you to populate your view with folders and files from the file system.

Initial Use and Appearance of Element Information

If you start ClearCase Explorer on a computer without any views defined, the Shortcut pane on the **Views** tab contains only the **General** page.

For any view that you create from within ClearCase Explorer that is not associated with a UCM project, ClearCase Explorer adds to the **General** page an icon with the view tag. (There is no way to tell ClearCase Explorer not to create an icon for a view that you own.) When ClearCase Explorer exits, it stores, as the default view, information about the view that you last worked in. When ClearCase Explorer starts again, it restores the default view.

Ongoing Use and Appearance of Element Information

Use the **View** menu to adjust the window appearance. Click the check boxes to toggle the display of window panes and bars. Click the **Refresh** commands to have ClearCase Explorer examine the system for the latest state information and redisplay the information in the Folder pane and Details pane. If you create a view outside ClearCase Explorer, clicking **View > Refresh View Shortcuts** finds the new view and adds a shortcut to the **General** page.

If you remove a view that you own, its shortcut in ClearCase Explorer is removed. To remove a shortcut for a view that you do not own, remove its view tag from the registry.

ClearCase Explorer Features for a New User

For new users, ClearCase Explorer offers an Information pane whose contents depend on the context of your work in a view. The Information pane presents workflow help topics that outline the stages required while working in different ClearCase environments.

As you work with ClearCase objects in a view, the Information pane displays contextual information about a selected element. Included is brief information about a selected element and its current state. For example, if you select a file, the Information pane shows two columns, one listing the details of your selection and the other offering, under the heading **About**, informative text or, under the heading **Uses**, tips on some of the operations that you can perform with the selected object.

In a view context in ClearCase Explorer, you can work in the Details pane where you can create a new file or directory. If you select the newly created file or directory, the Information pane tells you that you can add it to source control. If you add the file or directory to source control, the Information pane keeps the item selected and tells you that you can check it out.

After you learn the environment, you can hide the Information pane to reclaim the screen area.

Extended Namespace for Elements, Branches, and Versions

A version tree for an element has the same form as a standard directory tree (see Figure 6), which compares components of the version tree to components of a directory tree in extended namespace.

As a component of the version tree, the element is the root of the directory tree in the extended namespace. The element itself appears to be a directory, which contains a single subdirectory, corresponding to the **main** branch. (It can also contain some version labels.)

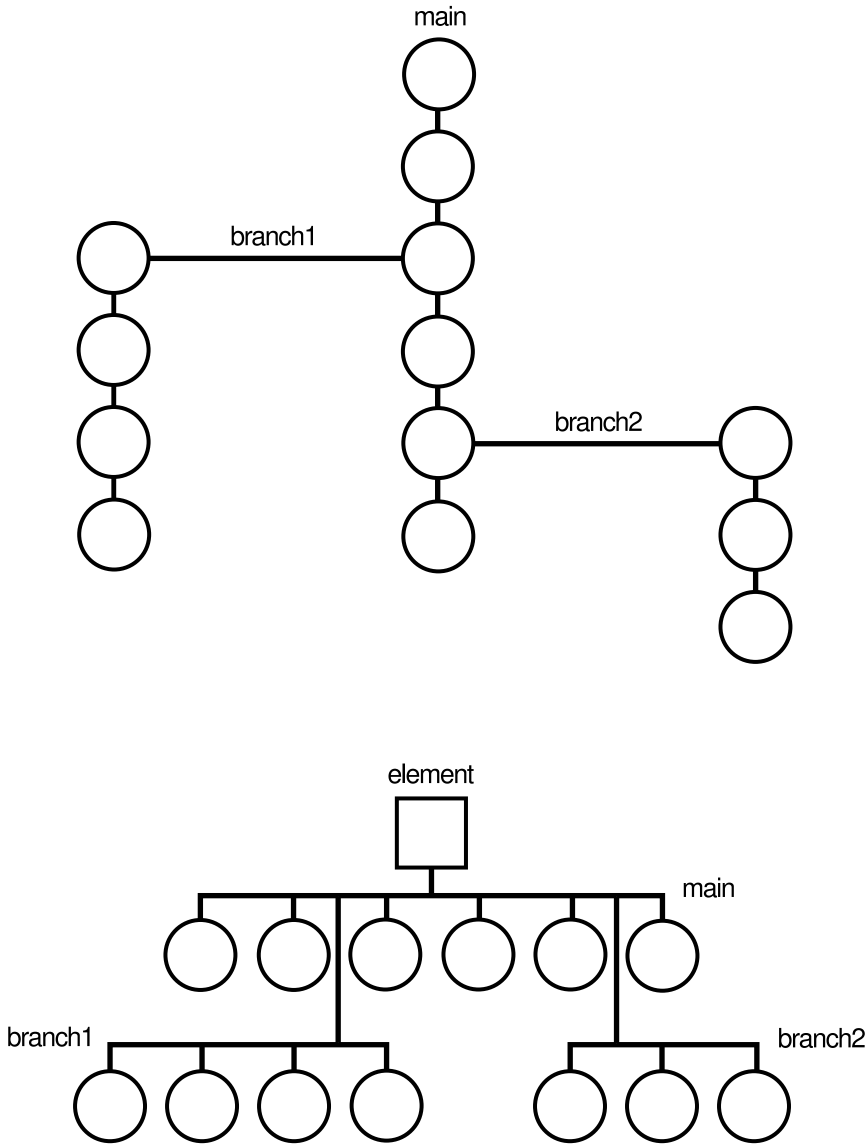
A branch in the version tree appears as a subdirectory in the extended namespace. As a directory, each branch can contain files (individual versions and version labels), directories (subbranches), and links (version labels).

A version in the version tree is a leaf name of the directory tree in the extended namespace. Each version of an element is a leaf of the directory tree. For a file element, the leaf contains text lines or binary data. For a directory element, the leaf contains a directory structure.

Accordingly, any location within the version tree of an element can be identified by a path in this extended namespace:

sort.c@@	*(specifies an element)*
sort.c@@/main	*(specifies a branch)*
sort.c@@/main/branch1	*(specifies a branch)*
sort.c@@/main/branch1/2	*(specifies a version)*
doctn/.@@/main/3	*(special case: extra component is required in top-level directory of VOB)*

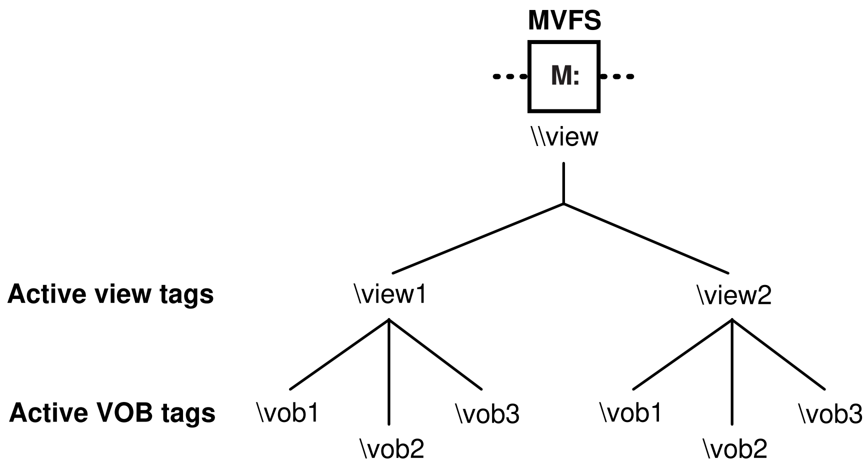
Figure 6 Version Tree and Extended Namespace



Dynamic View Access Model

All ClearCase elements in a dynamic view are accessed through the MVFS, which, by default, occupies drive M: on each ClearCase host. Each active view has a name called a *view tag* which appears in the root directory of M. Each active VOB has a path called a *VOB tag* which appears as a subdirectory under *each* active view, as shown in Figure 7.

Figure 7 ClearCase MVFS Namespace



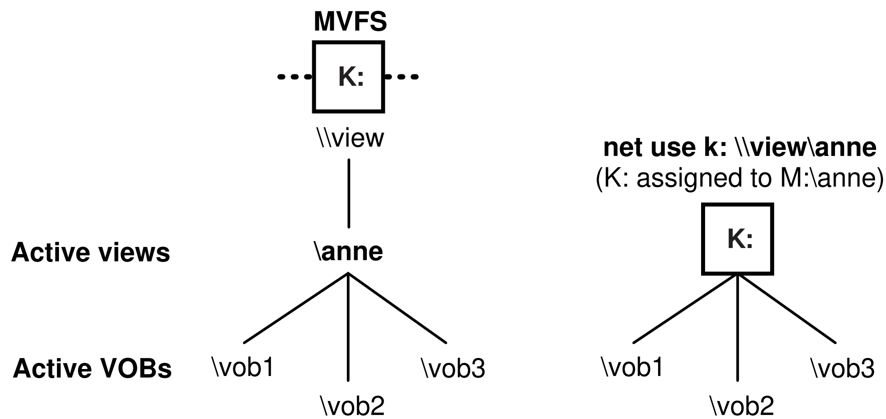
From drive M, you can access VOBs by using paths of the following form:

*view-tag**vob-tag**path-in-vob*

Typically, however, you do not work directly on drive M, but in the view root directory accessed by a shortcut in ClearCase Explorer. In Windows Explorer, you typically assign a drive to a view.

Figure 8 shows how the MVFS namespace looks from a drive assigned to a view in the ClearCase Explorer view shortcut, with the **net use** command, or by clicking **Tools > Map Network Drive** in Windows Explorer.

Figure 8 The MVFS Namespace from a Drive



From any drive, you can specify view-extended paths of the following form:

M**:***view-tag**vob-tag**path-in-vob*

If you move to drive M, you are in view-extended namespace, and all VOB access is by view-extended paths.

To eliminate any confusion that may result from unintentional use of view-extended paths when you are working at a command prompt, we recommend that you work from a drive letter assigned to a view. This permits you to use VOB paths of the following form in both **cleartool** and standard operating system commands:

drive-letter:*vob-tag**vob-object-path*

Furthermore, this approach is required if you want to share derived objects (DOs) between views at build time.

Setting Up a Work Environment

2

Usually when you set up a work environment, you establish a separate view for each development project to which you contribute. Setting up a view involves the following tasks:

- Choosing snapshot view or dynamic view
- Choosing a location and name
- Adding or modifying version-selection rules
- Selecting elements to load into snapshot views

Note: If you plan to access source files stored in UNIX VOBs, you may need to create your view in MS-DOS text mode, depending on how your shared source files handle line termination sequences. For more information, see *Accessing Views and VOBs Across Platform Types* on page 112.

If your project uses the base ClearCase-ClearQuest integration, you access one RationalClearQuest user database. Accessing a ClearQuest user database involves the following tasks:

- Choosing a master schema
- Setting up a profile file (.ini)

Starting the View Creation Wizard

The View Creation Wizard assists you in each step of setting up a view. Start ClearCase Explorer and the View Creation Wizard and use this chapter to complete the steps. For more details, click **Help**.

To Start ClearCase Explorer

Start ClearCase Explorer by doing one of the following:

- Click the Rational ClearCase shortcut on your desktop.
- If you did not install the shortcut, click **Start > Programs > Rational Software > Rational ClearCase > ClearCase Explorer**.

For information about ClearCase Explorer, see *About Rational ClearCase Explorer* on page 9.

To Start the View Creation Wizard

1 In the ClearCase Explorer Shortcut pane, click **Toolbox**. Then, click **Base ClearCase > Create View**.

2 The first step of the wizard asks whether you want to work on a UCM project. Click **No**, and then click **Next**. If you do want to work on a UCM project, see *Working in UCM*.

Choosing a Snapshot View or a Dynamic View

Depending on how your computer is configured, the View Creation Wizard may ask you to choose to create a snapshot view or dynamic view. As described in *ClearCase Views* on page 2, snapshot views load elements onto your computer; dynamic views use the *MVFS* to arrange VOB data into a directory tree. (Dynamic views may not be available on all platforms. For more information, see Help. To access Help, see *To Access Online Information* on page 1.)

Work in a snapshot view when any of these conditions is true:

- Your computer does not support dynamic views.
- You want to optimize build performance to achieve native build speed.
- You want to work with source files under ClearCase control when you are disconnected from the network that hosts the VOBs.
- You want to access a view from a computer that is not a ClearCase host.
- Your development project does not use the *build auditing* and *build avoidance* features of ClearCase.

Work in a dynamic view when any of these conditions is true:

- Your development project uses build auditing and build avoidance.
- You want to access elements in VOBs without copying them to your computer.
- You want the view to reflect changes made by other team members at all times (without requiring an *update* operation).

For more information, see the *Administrator's Guide* for Rational ClearCase.

Choosing a Location and Name

For a snapshot view, the View Creation Wizard prompts you to choose a location for the view. For a dynamic view, the wizard prompts you to choose a name, drive letter, and, the first time you create a dynamic view, a location for the *view storage directory*.

Snapshot View: Choosing a Directory

When creating a snapshot view, you must specify a directory into which ClearCase *loads* (copies) files and directories. When choosing a directory for the view, consider these constraints:

- The view root directory must be located on a disk with enough space for the files loaded into the view and any *view-private files* you add.
- Your organization may restrict where you can create a view. For example, you may be required to use a disk that is part of a data-backup scheme.
- If you want to access the view from other computers, it must be located in a shared directory.

If your makefiles or other files require absolute paths with a specific drive letter, assign the view to a drive letter. See *Assigning Snapshot Views to Drives* on page 106.

Under the Hood: A Snapshot View Storage Directory

Every snapshot view has a *view storage directory* in addition to the directory tree of source files that it loads from VOBs. ClearCase uses the snapshot view storage directory to keep track of such information as which files are loaded into your view and which versions are checked out to it. The view storage directory is for ClearCase administrative purposes only. Do not modify anything in it.

For every 1,000 elements loaded into the view, ClearCase uses about 400 KB of disk space for the view storage directory.

Locations for Snapshot View Storage Directories

Usually, your ClearCase administrator sets up a storage location, which is a directory on a ClearCase server host on UNIX or Windows. By default, ClearCase locates snapshot view storage directories there. If your ClearCase administrator sets up more than one storage location, ClearCase selects any one of these locations as the default when you create a view.

If your ClearCase administrator does not set up storage locations, by default, ClearCase software locates the view storage directory under the root directory of the snapshot view.

You can override these defaults. If your administrator sets up multiple storage locations, you can select one explicitly. If your ClearCase host is set up to store view storage directories (which happens when you install ClearCase), you can place the view storage directory under the root directory of the snapshot view. Or you can choose another location.

If you place the view storage directory under the root directory of the view, be aware of the following recommendations:

- Do not choose this configuration if you use the view when disconnected from the network. You can corrupt the data in the view storage directory if you disconnect it from the network while the *view_server* process of the view is running.
- Make sure that the view storage directory is accessible to any data backup schemes your organization institutes.

If you override the default and place the view storage location under a directory other than the root directory of the view, the directory must be below a shared network directory on a ClearCase host running Windows NT or Windows 2000 and must remain connected to the network. (A view process runs on the machine that physically stores the view storage directory, and only a ClearCase host running on Windows NT or Windows 2000 can run a view process.) The path for the directory must not use a Windows special share name, for example, the share that is designated by *drive*$ and allows an administrator access to a drive over the network. The directory cannot be on a removable storage device or on a laptop.

> **Note**: If you plan to work while disconnected from the network, or if your ClearCase host is not set up to store view storage directories, your administrator must set up storage locations.

To Override the Default Value for a Snapshot View Storage Location

1. When creating a view, on the step of the wizard that asks you to choose a location for a snapshot view, click **Advanced Options**.
2. In the Advanced View Options dialog box, do **one** of the following:
   - If **Use Server Storage Location** is selected and your administrator created multiple locations, ClearCase selects one for you. To select a different one, click the name of the location.

- If your computer is set up to store view storage directories and you want to locate the view storage directory in the root directory of the snapshot view or choose another location, select **Use explicit path**.

  Do not select this option if you plan to use the view while disconnected from the network.

3 If you select **Use explicit path**, do **one** of the following:

   - Accept the default path displayed in the View storage location box.
   - Edit the path in the View storage location box to specify a valid location.
   - Click **Browse** and select a valid location.

4 Click **OK** to return to the view location step of the wizard.

Choosing a View Name

Each view must have a descriptive name (called a *view tag*) that is unique within a network region. For dynamic views, the View Creation Wizard suggests a view tag based on the following convention: *username*_**view**. This name is designed to help you determine the owner and purpose of the view. Names like **myview** or **work** do not describe the owner or contents of the view; if you work with more than one view, such generic names can lead to confusion. Here are some suggested names:

pat_v1.4_cropcircle	Personal view for a user named Pat to develop source files for release 1.4 of the Cropcircle product
1.3_fix	Shared view for use in a particular bug-fixing task

The name of a view must be a simple name; that is, it must follow the format of a single file or directory name with no special characters or spaces.

Dynamic View: Choosing a Drive

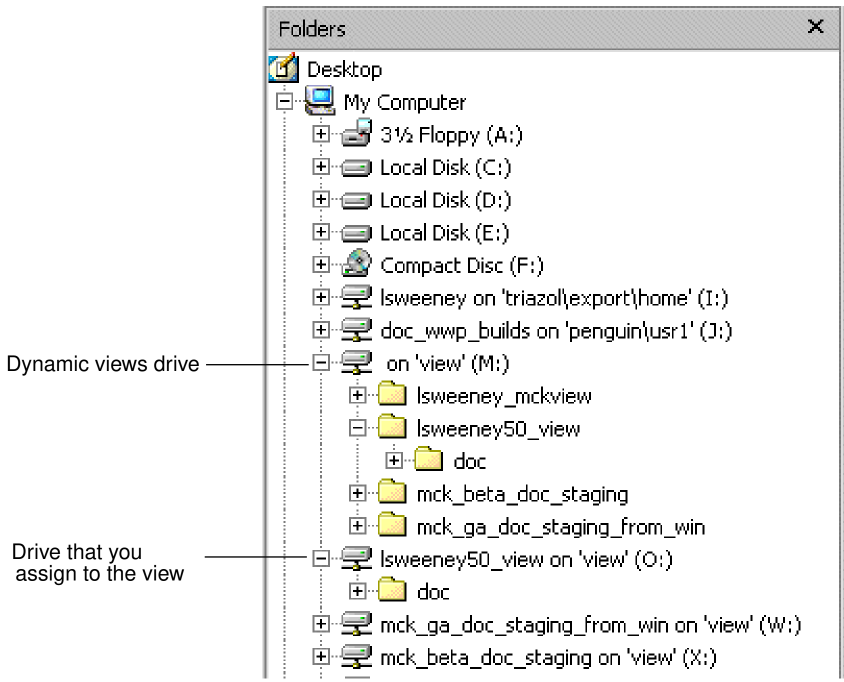
If your makefiles or other files require absolute paths, assign your view to a drive. When you use a wizard to create a view, ClearCase prompts you to assign the dynamic view to a drive. After creating a view, if you want to change a drive assignment or assign a drive to a team member's view, you can create or modify the assignment while adding a view shortcut to ClearCase Explorer.

In addition, you can create or modify drive assignments from Windows Explorer. Any changes you make to the drive assignments of a view outside ClearCase Explorer will invalidate the shortcut of the view in ClearCase Explorer.

Path Differences

In Windows Explorer, you can access any dynamic view that is started on your computer from the *dynamic-views-drive* (by default, drive M). However, when you access a view from the dynamic-views drive, its path includes one more component than when you access a view from an assigned drive letter (Figure 9).

Figure 9 Dynamic Views Started on a Host



For example, both of the following paths are available to an element lib.c in the lsweeney50_view dynamic view that is assigned to drive O:

```
M:\lsweeney50_view\doc\lib.c
O:\doc\lib.c
```

Dynamic View Storage Directories

If this is the first time you are setting up a dynamic view, ClearCase prompts you to choose a shared directory on your host as a location for the view storage directory. For dynamic views you create subsequently, ClearCase uses this location by default.

Every view has a view storage directory. For dynamic views, ClearCase uses this directory to keep track of which versions are checked out to your view and to store view-private objects. The view storage directory is for ClearCase administrative purposes only. Do not modify anything in it.

The size of the view storage directory depends on the following factors:

- Whether you use the **clearmake** or **omake** *build auditing* and *build avoidance* features
- The size and number of view-private files

For more information, see the *Administrator's Guide* for Rational ClearCase and the **clearmake** and **omake** reference pages in the *Command Reference*.

Choosing Locations for Dynamic View Storage Directories

If your computer is set up to store view storage directories, ClearCase reduces communication over the network by locating the view storage directory on your computer.

Consider the following restrictions when choosing a dynamic view storage directory location:

- The directory must be a subdirectory of a shared network resource on a ClearCase host (for example, on a Windows NT or Windows 2000 computer) or on a Network Attached Storage (NAS) device. View processes (specifically, *view_server* processes) run on the computer that physically stores the view storage directory or on the computer that can access a location on a supported NAS device which contains the view storage directory. And only ClearCase hosts on Windows NT or Windows 2000 can run view processes.

  The path for the directory must not use a Windows special share name. Special share names usually include the dollar sign ($), such as the drive$ share name that allows an administrator to gain access to a drive over the network. For example, \\bread\c$\view\pat_1.4_cropcircle.vws is not a valid path.

- To maintain data integrity, the view storage directory must remain connected to the network. For example, do not locate the view storage directory on a removable storage device.
- If you locate the view storage directory on a laptop and then disconnect the laptop from the network, all of the following restrictions apply:
  - You cannot use the dynamic view.
  - Team members who try to start your view from their hosts will receive error messages from ClearCase.
  - Any **clearmake** or **omake** process that attempts to wink in a derived object from your view will spend some amount of time trying to contact your view. If it cannot contact your view, it will not consider derived objects in your view as *winkin* candidates for 60 minutes. (You can change the amount of time by setting the CCASE_DNVW_RETRY environmental variable.) For more information, see the **clearmake** reference page.
- If your ClearCase administrator sets up storage locations (which are directories on ClearCase server hosts), you can locate your dynamic view storage directory in a storage location. However, for best performance, we recommend that you locate dynamic view storage directories on your local host.

We recommend that you make the view storage directory accessible to any data backup schemes your organization institutes.

To Choose a Location for a Dynamic View Storage Directory

1. On the step of the wizard that asks you to choose a name and drive for a dynamic view, click **Advanced Options**.
2. In the Advanced View Options dialog box, do **one** of the following:
   - Select **Use Explicit Path** and provide a UNC path to a shared directory on a ClearCase host on Windows NT. For best performance, we recommend this option.

     The first time you create a view with the wizard, ClearCase presents a template: `\\`*current-hostname*`\<Share>\`*view-tag*`.vws`

     To locate the view storage directory on your computer (recommended), replace `<Share>` with the name of a shared directory on your computer. For example, \\bread\view_storage\pat_v1.4_cropcircle.vws
   - Select **Use Server Storage Location**. If your ClearCase administrator created more than one location, ClearCase selects one for you. You can select a different one if you prefer.

Adding Version-Selection Rules

Development projects often require team members to add specific version-selection rules to the config spec for your view. This manual assumes that someone in your organization creates these rules, and you must either paste them into your config spec or add an inclusion rule so that your config spec includes them from a config spec available over the network. For information about creating version-selection rules, see *Managing Software Projects*.

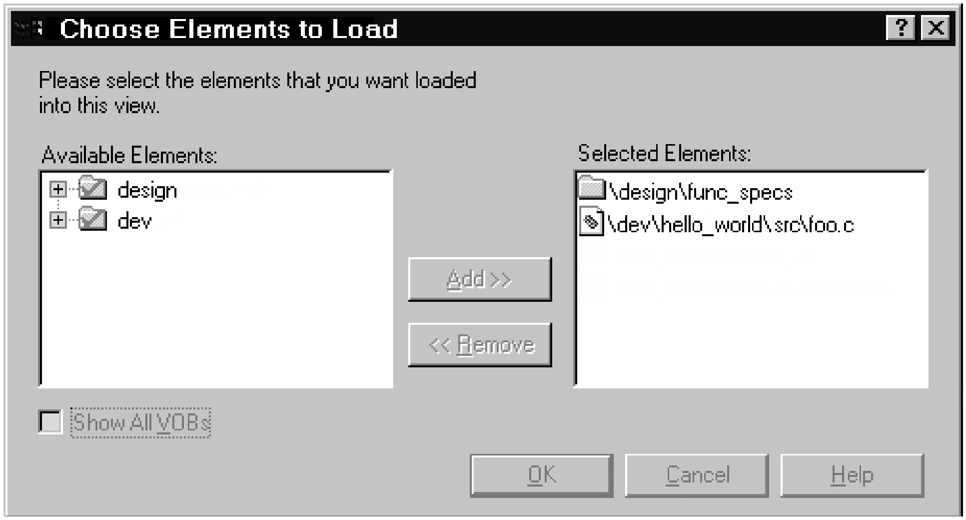
To Paste or Include Version-Selection Rules

1. In the View Creation Wizard, click **Finish**.
2. In the Confirm dialog box, click **Inspect Config Spec**; in the Config Spec Editor, click **Edit**.
3. In the Config Spec Editor, do any of the following:
   - Paste the rules from the Windows clipboard into the Config Spec Editor. Ask the author of the shared config spec whether you need to include any rules other than the ones you paste.
   - Type on its own line **include** *path-to-shared-config-spec*. Ask the author of the shared config spec whether you need to include any rules other than the include rule.
4. Click **OK**.
5. In the Confirm dialog box, do either of the following:
   - For snapshot views, take note of the path for the view root directory. This is the directory from which you access source files and directories. Then click **OK** to select elements to load into the view.
   - For dynamic views, take note of any drive that you assigned. Then click **OK**. By default, the view starts.

Snapshot View: Selecting and Loading Elements

After you set up version-selection rules for a snapshot view, the View Creation Wizard starts the VOB Namespace Browser (Figure 10). In the Choose Elements to Load dialog box, you select the files and directories to load into the view.

Figure 10 Choosing Elements to Load



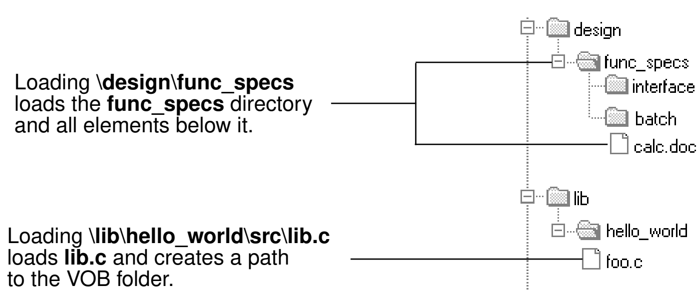
To Choose Elements

To choose the elements you want to load into your view, browse through the **Available Elements** list. To load an *element*, move it to the **Selected Entries** list. Keep the following in mind:

- If you load a directory element, all of its files and subdirectories are loaded into the view.
- If you load a specific file element, only that file is loaded into the view. To maintain the path to the file, ClearCase also copies into the view the empty directories leading up to the VOB directory.

Figure 11 illustrates the results of the load operation.

Figure 11 A View with Loaded Elements



For each element specified in the Entries to Load box, the View Creation Wizard adds a *load rule* to your config spec. For example, the Choose Elements to Load box shown in Figure 10 creates these two load rules:

```
load \design\func_specs
load \dev\hello_world\lib.c
```

Using these rules, ClearCase loads the directory element \design\func_specs and every element below it, along with the file element \dev\hello_world\src\lib.c. To maintain the path to lib.c, the View Creation Wizard creates the directory \dev\hello_world\src in the view.

Case-Sensitivity

If you are selecting elements from VOBs located on a UNIX host, you may encounter problems that are due to case-sensitivity. Because native file systems for UNIX are case-sensitive, it is possible to create two elements from a UNIX host whose names differ only in capitalization. For example, in a UNIX VOB, these two elements are distinct:

```
/design/func_specs/bas
/design/func_specs/Bas
```

However, Windows does not support case-sensitive file lookups and does not distinguish the two elements in the previous example. If you were to load these two elements, only one of them would have the correct data when copied into the view; duplicated files are reported as *hijacked*. (Hijacked files are discussed throughout Appendix A, *Working in a Snapshot View Without the Network*.)

The Rational ClearCase Help discusses case-sensitivity issues. We suggest that you not use mixed-case names for elements that you store in VOBs.

Setting Up for a New Development Project

If none of the files and directories for your development project are under ClearCase source control, click **OK** to close the Choose Elements to Load box. For information about adding elements to the VOB for a new development project, see *To Add Elements for a New Development Task* on page 97.

Loading Versions of Elements into a View

When you close the Choose Elements to Load box, ClearCase loads elements as follows:

1. It uses the *version-selection rules* to select one *version* of each *element* specified by a *load rule*.
2. It copies the version into the snapshot view.

3 It records which version it copies into the view.

As ClearCase loads files and directories into your view, it shows a progress indicator in the Snapshot View Update dialog box.

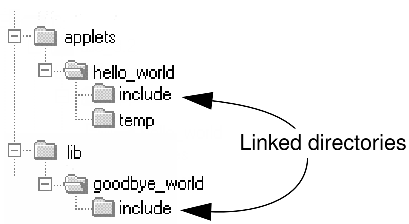
In the context of loading a snapshot view, links are treated as VOB links (those that point to objects inside the VOB) and non-VOB links (those that point outside the VOB). VOB links (both symbolic links and hard links) are followed. If a VOB link cannot be resolved, an error results. Non-VOB links are resolved, if possible. If they cannot be resolved, the load operation does not fail, but a warning is issued.

The complete list of files copied into your view appears in the Snapshot View Update window.

Under the Hood: VOB Links

A VOB link makes a file element or directory element accessible from more than one location in the VOB namespace. There are two kinds of VOB links: *symbolic links*, which are available for file and directory elements, and *hard links*, which are available for file elements only. We recommend that you use VOB symbolic links instead of VOB hard links whenever possible. In Figure 12, the directory element **include** is linked.

Figure 12 VOB Link



You use the **cleartool ln** command to create VOB links. For more information, see the **ln** reference page in the *Command Reference*.

Symbolic Links and Hard Links in Dynamic Views

In *dynamic views* (which use the MVFS, or multiversion file system), VOB links behave similarly to symbolic links or hard links in a UNIX file system: symbolic links point to a file or directory element in a different location, and hard links are alternative names for a single file element.

You cannot check out a VOB symbolic link; you must check out the symbolic link target.

When you check out a hard-linked element from a given path, ClearCase considers other paths for the element as "checked out but removed." That is, to prevent you from modifying the element from multiple paths, ClearCase executes standard checkout behavior at only one path (the one from which you entered the **checkout** command), but does not create view-private files at other paths. For information about standard checkout behavior, see the **checkout** reference page in the *Command Reference*.

Symbolic Links in Snapshot Views

Snapshot views created from a Windows host do not support links. ClearCase approximates VOB symbolic link behavior in the following ways:

- If a *load rule* selects a symbolic link, ClearCase copies the link target into the view at the path of the link.

  We recommend that you do not load a linked directory more than once in your view unless it is necessary for build purposes. Although ClearCase keeps track of multiply loaded elements accurately, you may become confused and modify the wrong copy of a file, losing information upon checkin. For more information, see *Caution: Losing Data Because of VOB Hard Links*.

- You cannot check out a file element from a symbolic link path; you must check out the link target. In ClearCase Explorer, the shortcut menu for a symbolic link includes the **Explore Link Target** command, which displays the parent directory of the link target. Use this command to find and check out the link target. When you check in the modified link target, ClearCase updates all associated symbolic links loaded in your view. Unlike directory link targets, you can check out a file link target only if it is loaded in your view.

- The **Add to Source Control** command (available from ClearCase graphic user interface tools) checks out, modifies, and checks in the link target directory whether you issue the command from a directory symbolic link or from the link target (and whether or not the target directory is loaded into the view).

  If you issue the **cleartool checkout** command from a symbolic link directory, you must use the following syntax:

  cleartool checkout .

  A **checkout** command issued for the current directory checks out the link target whether or not the target directory is loaded into the view.

  However, if you use the **cleartool checkout** *dirname* form of the command to check out a different directory, *dirname* must be a link target.

  When either you or ClearCase checks in a link target directory, ClearCase updates the associated symbolic links that are loaded in your view.

- If you *hijack* a file symbolic link, the Update Tool detects it. However, you cannot check out the hijacked symbolic link. To add your hijacked changes to the VOB, you must check out and modify the link target.

Hard Links in Snapshot Views

Each time a load rule selects a hard link, ClearCase loads the element into the view as a standard file.

Caution: Losing Data Because of VOB Hard Links

If you load multiple instances of a hard-linked element into a snapshot view, you must be careful to check out, modify, and check in only one instance of the file. When you check in a hard-linked file (or a file below a symbolic-linked directory), ClearCase updates all other instances in your view, which could result in loss of data if you modified multiple copies of the same file. (Note that, when updating instances of files because of a checkin, ClearCase renames any *hijacked* file to *filename*.keep before updating it.)

For example, the following sequence of events will lead to lost data:

1. You check out the hard-linked file src\util.h.
2. ClearCase removes the read-only attribute from util.h in the **src** directory only (which is the location from which you issued the **checkout** command).
3. You modify src\util.h but do not check it in.
4. Later, you lose track of which file you checked out. You then remove the read-only attribute and modify util.h in the **temp** directory.
5. You check in temp\util.h. Even though you checked out and modified src\util.h, ClearCase does not prevent you from checking in temp\util.h; with a VOB hard link, temp\util.h is just another name for src\util.h.
6. Any changes you made to src\util.h are lost upon checkin because ClearCase updates all copies of duplicated files when you check in an element. Note that ClearCase does not consider any copy of util.h to be hijacked (even if you change attributes), because you checked out the element in the VOB.

Setting Up ClearQuest User Database Defaults

If your project uses the base ClearCase-ClearQuest integration (see *The Base ClearCase-ClearQuest Integration* on page 7), the tasks that you perform can be associated with records in a ClearQuest user database. For each ClearQuest user database that you access, you can set up defaults for the following purposes:

- Controlling the appearance of the ClearQuest client display
- Receiving e-mail notification of ClearQuest user database changes
- Determining the print format of result sets
- Specifying start-up and operational actions

For information about accessing a ClearQuest user database, see *Logging On to a ClearQuest User Database* on page 48.

To Control the Display Appearance

You can control the appearance of the Result set and whether the SQL editor displays. To change the Result set appearance, do the following:

1. In the ClearQuest client, click **Edit > Grid Properties**. The Display Settings dialog box appears.
2. Select or clear an option. The **Preview** area shows the effect of the option setting.

   To save the setting in the ClearQuest user database profile, select **Save settings to profile**.
3. Click **OK**.

To control the display of the SQL editor, click **View** and select or clear **View SQL pane**.

The settings take effect immediately.

To Enable EMail Notification

Your project manager can define hooks in the ClearQuest user database to enable mail messages to be sent to users when certain state transitions occur in ClearQuest records. To provide the information for mail to reach you, do the following:

1. In the ClearQuest client, click **View > E-mail Options**. The Choose Email Provider page opens.
2. Select **Enable Email notification** to have mail messages sent to you and to have others notified of changes that you make.
3. In **Email Provider**, select the transport to use. Then click **Next**. The Configure Mail Server page opens.
4. Do one of the following:
   - If you selected the SMTP transport, in **Outgoing SMTP Server**, enter the name of the mail server that your ClearQuest administrator tells you to use.

In **Your Email Address**, enter the user and domain name of your mailbox (this address appears in the From field of messages triggered by your actions on a record and sent to others).

- If you selected the MAPI transport, in **Choose MAPI Profile**, select one from the list.

5 Click **Finish**. The Configure Mail Server page closes.

6 In the ClearQuest client, click **View > Change User Profile**. The User Profile dialog box opens.

7 In **Email**, enter the user and domain name of your mailbox.

This e-mail address is used in the Send field of messages directed to you. It differs from the similar field that you enter if you use the SMTP transport. This field is useful to specify the address of a group of users who share a common interest.

8 Click **OK**.

Notification is immediately enabled.

To Set Defaults for Printing a Result Set

To support printing, you can associate a report format with a record type, tailor the appearance of the Result set, and determine the location of date and paging information at the top and bottom of report pages.

Click **File > Print Preview** to see the effects of the settings that are described in this section.

To Associate a Format and Record Type

To associate a report format for printing with the record type of the ClearQuest user database that you have opened, follow these steps:

1 In the ClearQuest client, click **File > Print Record Setup**. The Print Record Setup dialog box appears with the name in **Record Type** inactive, and, if any format association is current, the path to that format.

2 Click **Associate Format**. The Select Report Format dialog box appears.

If you have a current association, click **Undo Association** and select another format.

3 Navigate to and click the report format entry and click **OK**.

To Change Result Set Print Appearance

To tailor the appearance of the result set in the print output, follow these steps:

1 In the ClearQuest client, click **File > Page Setup**. The Page Setup dialog box appears.

2 Select and clear the desired settings

To save the setting in the ClearQuest user database profile, select **Save settings to profile**.

3 Click **OK**.

To Set Headers and Footers

To set paging information in the print output, follow these steps:

1 In the ClearQuest client, click **File > Headers/Footers**. The Header/Footer dialog box appears.

2 Use the **Header** tab to define the alignment of text appearing at the top of each page and the **Footer** tab to specify the contents at the bottom of each page.

To save the setting in the ClearQuest user database profile, select **Save settings to profile**.

3 Click **OK**.

To Set Start-Up and Operational Defaults

When working in a ClearQuest user database, you can run queries when the database starts or can use optionally use the query wizard.

To Run Queries at Startup

In the ClearQuest client Workspace pane, right-click on a query to run and click **Run at Startup**. Note that you may run multiple queries at startup, but startup time increases with each query that runs.

To Use the Query Wizard

When you click **Query > New Query**, the Choose Record Type dialog box appears. You select a record type and click **OK**. If **Use Query Wizard** on the **Query** menu is clear, the **Display editor** tab in the Query builder pane becomes active so that you can create a query. If **Use Query Wizard** on the Query menu is selected, the ClearQuest Query Wizard starts.

To use the query wizard, click **Query** and select **Use Query Wizard**. The next time you create a new query, the Query Wizard starts.

Working in a View 3

This chapter guides you through the everyday tasks of managing source files from Rational ClearCase:

- Accessing files
- Checking out files
- Working with checkouts
- Canceling checkouts
- Checking in files

Accessing Files

The **Views** tab in ClearCase Explorer provides access to views (see *About Rational ClearCase Explorer* on page 9). By default, ClearCase Explorer creates shortcuts to all base ClearCase views that you own and places icons for the shortcuts on the **General** page of the **Views** tab. You can create your own pages to organize your base ClearCase views. The controls in the Folder pane enables you to navigate the current view.

Starting Dynamic Views

To access files from a dynamic view, you must start the view and activate the VOBs that contain your source files.

Starting the view also starts a *view_server* process, which maps the data in the VOBs that are activated on your computer into a directory tree. VOBs that you activate appear as subdirectories of a view. Then you browse the VOB as you would any other directory.

If you assign a view to a drive letter, ClearCase starts the view when you log on to your computer. If you do not assign a view to a drive letter but have a ClearCase Explorer shortcut to the view, ClearCase starts the view when you click the shortcut.

To start a dynamic view that was created from a UNIX host, you must first use the Region Synchronizer to import the view tag of the view into your Windows network region. Then, refresh the view shortcuts in ClearCase Explorer.

To Start Dynamic Views

1 In the ClearCase Explorer Shortcut pane, click **Views**. Then, click **General** or the page in which the shortcut to the dynamic view resides.

2 Click the shortcut for the view.

The Folder pane activates. The view tag appears as the top-level folder in the Folder pane with folders below for the active VOBs (see *About Rational ClearCase Explorer* on page 9). If you do not have a shortcut, refresh the view shortcuts (if you own the view or created the view tag). To access other dynamic views from ClearCase Explorer (for example, to access a team member's view), add a view shortcut to the **Views** tab. For information about creating or modifying shortcuts in ClearCase Explorer, see ClearCase Explorer Help. To access Help, see *To Access Online Information* on page 1.

The ClearCase Explorer title bar shows the location of the view in the namespace of your computer. Any command you issue from ClearCase Explorer that requires a path uses the path displayed in the title bar.

If you are unable to start a dynamic view that is on another host, check with your administrator to make sure that you can access the view storage directory of the view. For more information, see the *Administrator's Guide* for Rational ClearCase.

To Activate VOBs

To access files from a dynamic view, you must activate the VOBs that contain your source files.

1 On the Windows desktop, click **Start > Programs > Rational Software > Rational ClearCase > Administration > Mount VOB**.

2 In the Mount dialog box, select the VOBs that contain your source files.

3 Select **Reconnect at Logon** to activate the VOBs when you log on.

4 Click **Mount**.

5 In ClearCase Explorer, click **View > Refresh** to see the activated VOBs as folders in the Folder pane.

For information about deactivating VOBs, see *Activating or Deactivating VOBs in Dynamic Views* on page 105.

Accessing Snapshot Views

To access files from a snapshot view, use the view shortcut in ClearCase Explorer.

1 In the ClearCase Explorer Shortcut pane, click **Views**. Then, click **General** or the page in which the shortcut to the snapshot view resides.

2 Click the shortcut for the view.

If you do not have a shortcut, either refresh the view shortcuts (if you own the view) or add a view shortcut (if someone else owns the view; see *To Access Another's Snapshot View from ClearCase Explorer* on page 109).

The view tag appears as the top-level folder in the Folder pane with folders below for the elements loaded in the view (see *About Rational ClearCase Explorer* on page 9). The ClearCase Explorer title bar shows the location of the view in the namespace of your computer. Any command you issue from ClearCase Explorer that requires a path uses the path displayed in the title bar.

If you are unable to access a snapshot view that is on another host or its storage location is on another host, check with your administrator to make sure that you can access the view directory or view storage location of the view. For more information, see the *Administrator's Guide* for Rational ClearCase.

If you plan on working disconnected from the network, follow the guidelines in Appendix A, *Working in a Snapshot View Without the Network*.

Accessing Views from Windows Explorer

This section describes how to access snapshot views and dynamic views from Windows Explorer.

Accessing Snapshot Views from Windows Explorer

A snapshot view is a directory tree in a standard file system (plus some hidden, administrative files). You can access it through Windows Explorer as you would access any other directory tree in a file system. For information about assigning a snapshot view to a drive letter, see *Using the subst Command for Snapshot View Access* on page 108.

Accessing Someone Else's Snapshot View from Windows Explorer

You can access someone else's snapshot view as you would access any other directory on another computer. If the owner of the directory has shared the directory and set up the proper permissions, you can use Network Neighborhood to access the view. If you want to perform ClearCase operations in the view, see *To Register a Snapshot View by Using Windows Explorer* on page 109.

Accessing Dynamic Views from Windows Explorer

You can access any view that you have started on your computer from the *dynamic-views-drive* (by default, drive M) in Windows Explorer. If you assign the view to a drive letter, you can also access it from the drive letter in Windows Explorer. For information about assigning dynamic views to drive letters, see ClearCase Help.

Checking Out Files

To modify files under ClearCase control, you must check them out. (Placing files and directories under source control is a separate procedure; see *Adding Files and Directories to Source Control* on page 91.) If you work in an environment with the base ClearCase-ClearQuest integration, you may have to perform additional steps (see *Checking Out Files in a VOB Enabled for ClearQuest* on page 47).

To Check Out Files

1 In ClearCase Explorer, navigate to the directory that contains the files you want to check out. Then select the files.

2 Right-click one of the selected files and click **Check Out**. The Check Out dialog box opens.

3 In **Comment**, provide a description of the changes you plan to make.

4 Determine whether you want a *reserved* or *unreserved checkout* (see *Reserved and Unreserved Checkouts* on page 44). Do **one** of the following.

   - Select **Reserved** and optionally select **Unreserved if already reserved**.
   - Clear **Reserved**.

5 Click **OK**.

Using the Open Dialog Box

The Open dialog box that many applications use is actually part of ClearCase Explorer. As illustrated in Figure 13, right-clicking a loaded version of a ClearCase element in the Open dialog box displays a ClearCase shortcut menu.

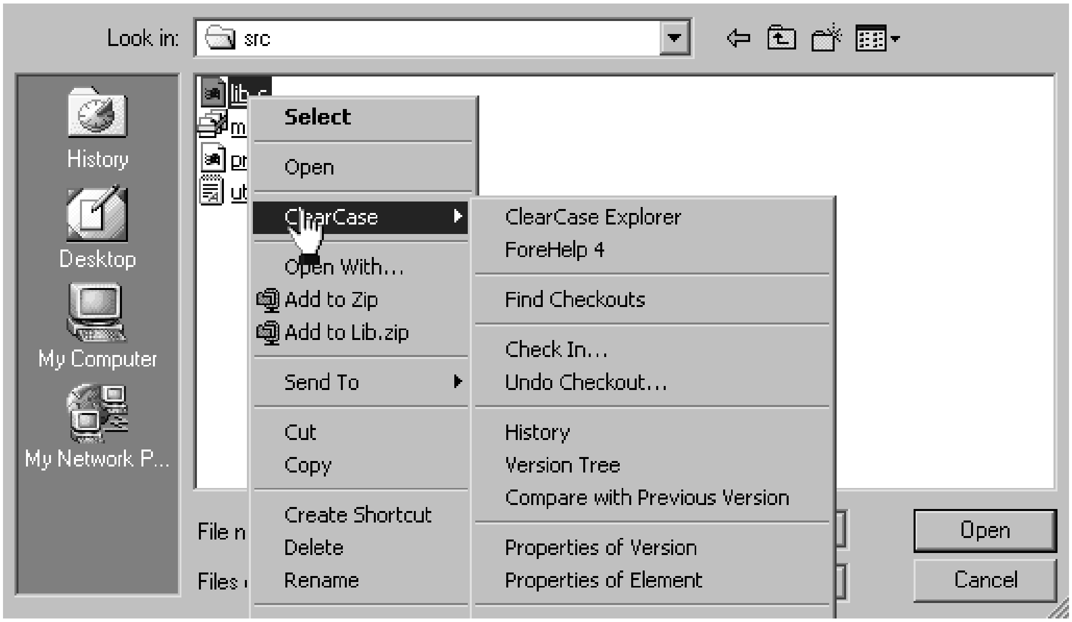
To check out a file from an Open dialog box:

1 In an application (for example, Microsoft Notepad), click **File > Open**.

2 In the Open dialog box, access the file or directory that is under source control.

3 Right-click the file; then click **ClearCase > Check Out**.

Figure 13 ClearCase Shortcut Menu from the Open Dialog Box



Checking Out Directories

Directories, as well as files, are under ClearCase source control, yet you rarely need to check out a directory explicitly. ClearCase checks out and checks in the parent directory of a file when you add the file to source control.

What does it mean for a directory to be under source control? In a version-controlled directory, ClearCase creates a new version of the directory when you add or rename a file element under source control. Having versions of directories can be helpful if, for example, you rename a source file used in a particular release and then modify your makefile to reflect this change. If you need to rebuild a previous release, you can set up your view to select the version of the directory that contains the file under its previous name.

Note: When you issue commands from a command-line interface (CLI), such as an MS-DOS command prompt, ClearCase does not check out directories automatically. When using a CLI to change a directory element, you need to check out the directory explicitly. For more information about checking out files and directories from a CLI, see the **checkout** reference page in the *Command Reference*.

Reserved and Unreserved Checkouts

In some version-control systems, only one user at a time can reserve the right to create a new version. In other systems, many users can compete to create the same new version. ClearCase supports both models by allowing two kinds of checkouts: reserved and unreserved.

The view with a reserved checkout has the exclusive right to check in a new version for a given development project. Many views can have unreserved checkouts. An unreserved checkout does not guarantee the right to create the successor version. If several views have unreserved checkouts, the first view to check in the element creates the successor; developers working in other views must merge the checked-in changes into their own work before they can check in. The development policy of your organization may determine whether to check out reserved or unreserved.

Figure 14 illustrates checked-out versions created by reserved and unreserved checkouts, and the effects of subsequent checkins.

Another kind of checkout is an unreserved, nonmastered checkout, which can be used only in a replicated VOB (created with Rational ClearCase MultiSite). For more information about this kind of checkout, see *Sharing Control of a Branch with Developers at Other Sites* on page 85.

To Change the Status of a Checked-Out Version

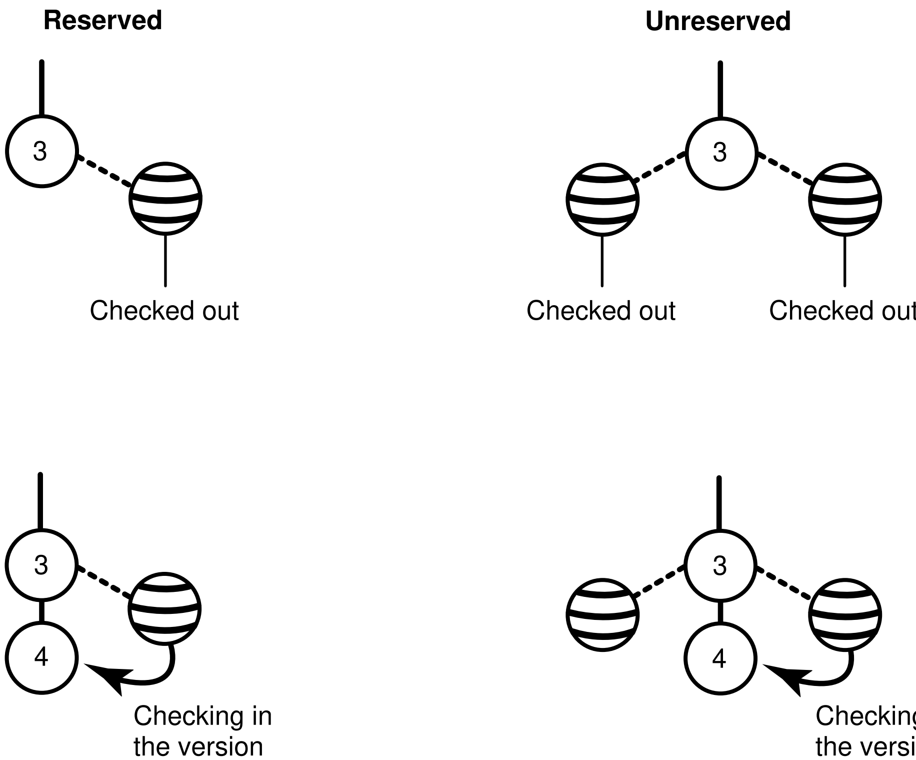
1. In the ClearCase Explorer Details pane, select a checked-out version.
2. Click either **Tools > Reserve** or **Tools > Unreserve**.

To Set the Default for Reserved or Unreserved Checkouts

Note: Your ClearCase administrator can make this option unavailable to you.

1. In ClearCase Explorer, click **Tools > Options**. The Options dialog box appears.
2. Click **ClearCase Options**. The ClearCase User Options dialog box appears.

Figure 14 Resolution of Reserved and Unreserved Checkouts



3 In the Check Out box, do **one** of the following:

- To make reserved checkouts the default setting, select **Reserved**.

  With this selection, you can also select **Unreserved if already reserved** to check out unreserved by default if someone else has a reserved checkout for the same element. If you do not select this check box, attempts to reserve a reserved checkout fail.

- To make unreserved checkouts the default setting, clear **Reserved**.

For more information about unreserved, nonmastered checkouts, see *Setting the Default for Nonmastered Checkouts* on page 89.

Under the Hood: What Happens When You Check Out a File or Directory

Because a snapshot view contains copies of files and directories, and a dynamic view provides access to data in VOBs, ClearCase follows different procedures for checking out from the different view types.

Checking Out From a Snapshot View

When you use the graphic user interface to check out a file or directory from a snapshot view, the request is handled as follows:

1 It gathers the following information:

- The version currently loaded in the view
- The version selected by the config spec
- The latest version in the VOB

2 If the version of a file in your view is not the latest in the VOB, ClearCase asks you to specify which version to check out. For directory elements, ClearCase requires you to check out the version of the directory currently loaded in your view.

   The version in your view will not be the latest in the VOB if either of these conditions exist:

   - Someone else has checked in a new version since you last updated your view.
   - The config spec of your view selects versions based on a label or a time rule, and the latest version in the VOB falls outside those parameters (Figure 15).

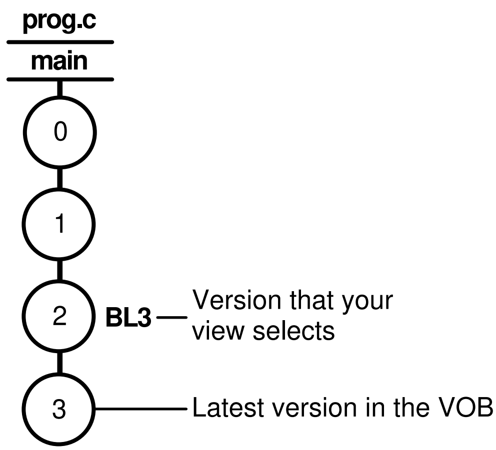
3 If you check out a version other than the one currently loaded in your view, ClearCase loads the checked-out version into your view.

4 ClearCase notifies the VOB which version of the element you checked out.

5 For files, it removes the **Read-Only** attribute. For directories, it allows you to add new elements to source control.

For information about checking out VOB links in a snapshot view, see *Under the Hood: VOB Links* on page 32.

Figure 15 Selecting the Nonlatest Version of an Element



Checking Out From a Dynamic View

When you use the graphic user interface to check out a file from a dynamic view, ClearCase handles the request as follows:

1 If the version-selection rules for your view do not select the latest version in the VOB, ClearCase prompts you to choose a version to check out.

   Your view may not select the latest version in the VOB if, for example, your config spec selects versions based on labels or time rules (Figure 15).

   For information about checking in a version that is not the latest on the branch, see *Merging the Latest Version with Your Changes* on page 55.

2 ClearCase notifies the VOB which version of the element you checked out.

3 For files, ClearCase creates in the view an editable view-private file, which is a copy of the checked-out version. For directories, it allows you to use add new elements to source control.

Checking Out Files in a VOB Enabled for ClearQuest

If the VOB in which you access versions is set up for the base ClearCase-ClearQuest integration, you may have to associate the version on which you are working with a ClearQuest record. For more information, see *The Base ClearCase-ClearQuest Integration* on page 7.

Logging On to a ClearQuest User Database

The first time that you attempt to check out a file from or check in a file to a VOB enabled for the base ClearCase-ClearQuest integration, you are prompted to log in to ClearQuest. Specify a ClearQuest user ID, password, and database name. ClearQuest keeps its own database of user IDs and passwords. Make sure that your ClearQuest administrator has added a user ID and password to the ClearQuest user database for you.

After you log in to ClearQuest, you can use the integration to complete your checkout and checkin operations (see *Using the Modified Graphic User Interfaces*). The integration stores the user ID and an encoded version of the password in a file named .cqparams, which it creates in platform-specific areas. On Windows computers, the path to the file is:

```
<windows>\Profiles\<login>\Application Data\Rational\Cqcc\
```

Thereafter, the integration uses the user ID and password in .cqparams instead of prompting you for them. If you change your password or connect to a different ClearQuest user database, the next time you check out or check in a version from that VOB, the integration displays an error message and prompts you to reenter the user ID and password. The integration updates the .cqparams file with the new information.

The Base ClearCase-ClearQuest Integration Interfaces

If you are logged in to a ClearQuest user database (see *Logging On to a ClearQuest User Database* on page 48) and check out or check in a file in a VOB enabled for ClearQuest, you see a modified interface, one for a graphic user interface or one for a command line interface. If you use ClearCase Explorer or Windows Explorer, the Association dialog window opens (see *Using the Modified Graphic User Interfaces* on page 48). If you use a **cleartool** command line interface (CLI), you see a numbered menu (see *Using the Modified Command Line Interface* on page 49).

Using the Modified Graphic User Interfaces

For the modified graphic user interface, one integration is based on a Perl trigger. The Association dialog window displays the result set of a standard ClearQuest query, usually showing all open change-request IDs currently assigned to you. If local policies allow, you can select alternative site-specific or ClearQuest-supplied queries to run.

Select an entry in the result set and click **Associate** to create an association between the versions you are checking out and the change request. If multiple selections are allowed on your project, select multiple entries. With at least one association, click **OK** to continue the checkout. To close the dialog box without making a change, click **Cancel**. For more information, click **Help** in the window.

Some systems use an alternative integration based on a Visual Basic trigger to provide a graphic user interface. (This interface lacks the **Database** and **Record Type** buttons that the Perl-base interface has.) You use the Associate Change Requests dialog box to associate a ClearCase version with one or more ClearQuest change requests and to remove existing associations. To change your stored user name, password, or ClearQuest user database name, use the Registry Editor to delete the key similar to the following:

```
HKEY_CURRENT_USER\Software\Rational
Software\ClearQuest\2003.06.00\Common\CQIntSvr
```

The integration prompts you for the information the next time you checkin or checkout a version.

Using the Modified Command Line Interface

For the modified **cleartool** command line interface, there are either text menus with numbered options or a **clearprompt** window showing the same menus. The options in the menus are shown in Table 1.

Table 1 Base ClearCase-ClearQuest Integration Options (Part 1 of 2)

Option	Description
OK - commit associations	Make the requested associations and exit.
CANCEL	Exit without making any changes and cancel the related ClearCase operation.
HELP	Display this text.
REVIEW Associations	Shows currently selected associations and allows you to delete one or more items from the list.
QUERY – Select from ClearQuest Query	Displays the contents of a query that your ClearCase administrator defines to show defects that are appropriate. Depending on your local policy, you select one or multiple items.
QUERYNAME	Shows the current query in Query. If this option appears, you see a list of alternative queries either defined by the configuration file (LOCAL) or in the ClearQuest user database itself. Your site administrator determines which queries appear.

Table 1 Base ClearCase-ClearQuest Integration Options (Part 2 of 2)

Option	Description
TYPEIN – Enter Associations from Keyboard	Allows you to enter one or more change-request IDs directly.
DATABASE	Shows the current ClearQuest user database name; allows you to change to a different database.
RECORD TYPE	Shows the current record type with which you are working; for example, a defect. Allows you to change to a different record type if multiple entities are supported by the current ClearQuest user database.
PATHNAME	Shows the full path for the version that you are checking in or checking out. Select this item to see more information.

To run an option:

- In the text menu, type the number of the entry and press ENTER.
- In the **clearprompt** window, select an option and click **OK**.

All menus accept a single choice, to which you can enter a number. TYPEIN allows multiple change-request IDs, separated by a space or a comma. The IDs can be full names (for example, SAMPL00056789) or numbers (for example, 56789).

To dismiss the menu without making a choice, simply press ENTER.

Using the Menu to Associate a Checkout with a ClearQuest Entity

If you use the command line interface and are required to associate versions with change requests, use the options from Table 1 to enter change-request IDs as follows:

- Use the QUERY option to see a list of all open change-request IDs currently assigned to you and select one or more items from the list to make associations.
- Use the TYPEIN option to enter one or more change-request IDs with which to associate the version that you are checking out.
- Use the REVIEW option to list and possibly delete current associations.
- If the association is optional and you do not want to specify a change request, enter the OK option and click **Yes** (**clearprompt**) or press ENTER (text menu).
- To cancel the checkout operation, use the CANCEL option or click **Abort**.

- To display help text, use the HELP option.

After you specify your options, use the **OK** option to create or update the associations that you specified and complete the checkout operation.

Working with Checkouts

After you check out a version, you do not need to interact with ClearCase until you are ready to check in. However, some ClearCase tools can help you with the following tasks:

- Viewing the history of an element
- Comparing versions of elements
- Tracking checked-out versions

Viewing the History of an Element

The History Browser displays the history of an element modifications, including version-creation comments (entered when someone checks out or checks in an element).

To View Element History

In the Details pane in ClearCase Explorer, right-click an element and click **History**.

Comparing Versions of Elements

As you modify source files, you may want to compare versions to answer such questions as these:

- What changes have I made in my checked-out version?
- How does my checked-out version differ from a particular historical version or from the version being used by one of my team members?

To Compare with a Predecessor

In the Details pane in ClearCase Explorer, right-click a file and click **Compare with Previous Version**.

To Compare with a Version Other Than the Predecessor

1 In the Details pane in ClearCase Explorer, right-click a file and click **Version Tree**.

2. In the Version Tree Browser, right-click a version and click **Compare > with Another Version**.

   The cursor changes to a target icon.

3. Click the version you want to compare.

To Compare Any Two Files

If you want to compare any two files, you can use the **Diff Merge** utility from a **Send To** menu. First, create a shortcut in your **SendTo** folder. Use Windows Explorer to navigate to the **SendTo** folder, right-click in the details pane and click **New > Shortcut**. In the **Create Shortcut** dialog box, enter **cleardiffmrg.exe** in the **Type the location of the item** box. No path is necessary. Then stop and restart ClearCase Explorer.

After you create the entry in your **Send To** menu, right-click any file in the Details pane in ClearCase Explorer (also Windows Explorer or any application with a **Send To** menu) and click **Send To > cleardiffmrg**. Then, in the **Open a Second File to Compare with** dialog box, navigate to the second file and click **Open**. Diff Merge runs. This method can be used anywhere in the file system, not just within a view.

Tracking Checked-Out Versions

Depending on how you work, you may forget exactly how many and which files are checked out. To list all the files and directories you currently have checked out to your view:

1. In the ClearCase Explorer, right-click a folder for a versioned directory and click **Find Checkouts**.
2. In the Find Criteria dialog box, edit the path from the root directory of the snapshot view in the **Search folder** box.
3. Under **Include additional folders?**, select **Include subfolders**.
4. Click **OK**.

Prototype Builds

Typically, when you are developing source files for a project, you want to perform prototype builds to test your modifications. If your organization uses **clearmake** or **omake**, you can use these ClearCase build tools for your prototype builds; however, the *build auditing* and *build avoidance* features are available only from dynamic views.

For more information, see *Building Software* and the **clearmake** or **omake** reference pages in the *Command Reference*.

Canceling Checkouts

If you check out a file but do not want to check in your changes or want to start with a fresh copy, you can cancel the checkout as follows:

1. In the Details pane in ClearCase Explorer, select one or more checkouts. Then right-click with the pointer over the selection and click **Undo checkout**.
2. Click **Yes** in the Confirm Undo Checkout dialog box. You can choose to save any of your changes in the file *filename*.keep.

Under the Hood: Canceling Checkouts

When you cancel the checkout of a file element, ClearCase handles the request as follows:

1. It prompts you to rename the file in your view to *filename*.keep.
2. It notifies the VOB that you no longer have the version checked out in your view.
3. In a snapshot view, it copies from the VOB the version that was in your view when you performed the checkout operation.

   In a dynamic view, it uses the version-selection rules of the config spec to select a version.

If you work in an environment with the base ClearCase-ClearQuest integration, any associations with ClearQuest change requests you may have made at checkout (see *Checking Out Files in a VOB Enabled for ClearQuest* on page 47) are canceled if you cancel the checkout.

Canceling Directory Checkouts

Although you rarely need to check out a directory explicitly, if you do and then cancel the checkout, ClearCase notifies the VOB that you no longer have the version of the directory checked out to your view. ClearCase does not prompt you to rename a canceled directory checkout to *directory-name*.keep.

If you cancel a directory checkout after changing its contents, any changes you made with **cleartool rmname**, **mv**, and **ln** are lost. Any new elements you created with **mkelem** or **mkdir** become orphaned. (When you create elements from the graphic user interface with the **Add to source control** command, ClearCase checks out the directory, adds the element, and checks in the directory, avoiding the creation of orphaned elements.) ClearCase moves orphaned elements (and any data that exists in the view at the path of the new element) to the lost+found directory in the VOB under names of this form:

element-name.UUID

In such cases, **uncheckout** displays this message:

```
cleartool: Warning: Object "prog.c" no longer referenced.
cleartool: Warning: Moving object to vob lost+found directory as
"prog.c.5f6815a0a2ce11cca54708006906af65".
```

In a snapshot view, ClearCase does not remove *view-private objects* or start the update operation for the directory in the view. To return the directory in your view to its state before you checked it out, you must start the Update Tool. For information about starting the Update Tool, see *To Start the Update Tool* on page 61.

In a dynamic view, ClearCase does not remove view-private objects, but it does revert the view to its previous state.

To Move and Delete Orphaned Elements

To move an element from the lost+found directory to another directory within the VOB, use the **cleartool mv** command. To move an element from the lost+found directory to another VOB, use the **relocate** command. For more information about moving elements to another VOB, see the **relocate** reference page in the *Command Reference*.

To permanently delete an element in the lost+found directory, take note of the name of the orphaned element and use this command:

cleartool rmelem *VOB-path*\lost+found*orphaned-element-name*

For example, from a dynamic view:

cleartool rmelem \guivob\lost+found\prog.c.5f6815a0a2ce11cca54708006906af65

From a snapshot view:

cd c:\pat_v1.4_cropcircle_sv
cleartool rmelem guivob\lost+found\prog.c.5f6815a0a2ce11cca54708006906af65

> **Note**: In a snapshot view, ClearCase treats the lost+found directory, which is located immediately below the root directory of a VOB, as any other directory. To load the directory in your view, you must use a load rule that specifies either the parent directory of the element or the directory itself. However, as with any other directory in a snapshot view, you do not need to load the lost+found directory to issue ClearCase commands for elements in the directory.

To Cancel the Checkout of a Deleted File

If you check out a file element and then delete that file from your view (by using, for example, the **Delete** command in ClearCase Explorer), use this procedure to cancel the checkout:

1 In the ClearCase Explorer Folder pane, right-click the directory containing the deleted file and click **Find Checkouts**.

2 Complete the Find Criteria dialog box.

3 In the Find Checkouts window, select the file whose checkout you want to cancel and click **Undo Checkout**.

Checking In Files

Until you check in a file, ClearCase has no record of the work in your view. Checking in a file or directory element creates a new version in the VOB, which becomes a permanent part of the history of the element. We recommend that you check in a file or directory any time you want a record of its current state.

Ideally, the development strategy of your organization isolates your checked-in work from official builds and requires you to merge your work to official project versions at specified intervals.

To Check In Files

1 In the Details pane in ClearCase Explorer, select one or more files.

2 Right-click a selected file and click **Check In**.

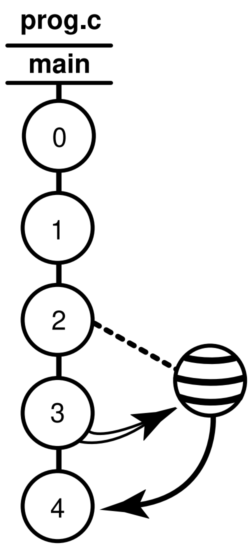
3 The Check In dialog box displays the comments you entered when you checked out the file. You can reuse these comments or modify them.

Merging the Latest Version with Your Changes

If the version you checked out is not the latest version in the VOB and you try to check in your modifications, ClearCase requires you to merge the changes in the latest version into the version checked out in your view (Figure 16).

In Figure 16, version 2 of prog.c is the one that you checked out. Before you check in your modifications, someone else checks in version 3 of prog.c. When you check in your modifications, ClearCase tells you that the version you checked out is not latest on the branch. (For more information about situations in which you may have to merge before checking in, see *Under the Hood: What Happens When You Check Out a File or Directory* on page 45.) Note that the reserve status of the checkout is not relevant to whether your modifications can be checked in.

Figure 16 Merging the Latest Version and Your Checkout



You need to merge the latest version in the VOB (prog.c@@/main/LATEST) to the version in your view before you can check in your modifications. This merge creates a version that reconciles modifications made in the latest version with your modifications. Then, when you check in the merge results, the system sees the merge arrow from version 3 to your checked-out version containing the merge results. The checkin creates a version 3 successor, version 4 of prog.c.

To Merge the Latest Version

When you issue the **Check In** command for a version that is not the latest on the branch, you are prompted to merge. If you choose to merge, ClearCase attempts to merge automatically, starting the Diff Merge tool if it needs your input to complete the merge. For information about using Diff Merge, see ClearCase Help. After the merge, ClearCase prompts you to check in the file.

Under the Hood: Checking In Files

The steps ClearCase follows when you issue the **checkin** command vary depending on the kind of view you use.

Checking In From a Snapshot View

When you issue a **checkin** command from a snapshot view, ClearCase handles the request as follows:

1. It copies your modifications to the VOB as a new version.

   The version you check in remains in the view, regardless of the config spec of the view.

2. It removes write permission for the file.

For any other instance of the file loaded into a snapshot view, ClearCase copies the new version from the VOB into your view. (If your load rules specify an element that appears in more than one VOB location, the element is copied into each of the appropriate locations in the directory tree of your view.)

When you check in a symbolic-linked directory from a snapshot view, ClearCase does not update any other instances of the directory loaded in your view. As you add file elements to source control, ClearCase adds a copy of the element to all instances of a parent directory loaded in your view.

Checking In From a Dynamic View

When you issue the **checkin** command from a dynamic view, ClearCase handles the request as follows:

1. It copies your modifications to the VOB as a new version.
2. It uses the version-selection rules in the config spec to select a version from the VOB. If the config spec selects a version other than the one you checked in, ClearCase displays a message. ClearCase may select another version if, for example, your view selects versions based on labels or time rules.
3. It removes the view-private file and provides transparent access to the version checked in to the VOB.

Checking In Files in a VOB Enabled for ClearQuest

If you use the base ClearCase-ClearQuest integration (see *Checking Out Files in a VOB Enabled for ClearQuest* on page 47), the version you are checking in must be associated with at least one change request; otherwise, the checkin cannot proceed. When you check in the version, the base ClearCase-ClearQuest integration displays those change-request IDs whose associations you made during checkout. You can do the following:

- Keep the same change-request IDs.

- Delete some or all of the change-request IDs.
- Add new change-request IDs.

The base ClearCase-ClearQuest integration creates associations for new change-request IDs that you add, removes associations for change-request IDs that you delete, and updates information on existing ones.

Using the Association Dialog Window

If you use ClearCase Explorer or Windows Explorer, the Association dialog window appears. You can do the following:

- Click **Browse** to navigate to and run another query.

  In the result set, select entries and click **Associate** to add the selections to **Associated Records**.

- In **Associated Records**, select an entry and click **Disassociate** to delete the current association.

In the Association dialog window, click **OK** to continue the checkin. For more information, click **Help**.

Using the Command Line Interface Options

If you use a **cleartool** command line interface (CLI), you can do the following with the options in Table 1 on page 49:

- Use the QUERY option to see a list of all open change-request IDs currently assigned to you.
- Use the REVIEW option to list and possibly delete current associations.
- Use the TYPEIN option to enter one or more change-request IDs with which to associate the version that you are checking in.

If the associations are correct, use the OK option to continue the checkin.

View the Versions for a Change Request from ClearQuest

To view the versions associated with a ClearQuest change request:

1. In ClearQuest, run a query to find the desired set of change requests.
2. In the query result set, select a change request to display its Record form.
3. On the Record form, click the **ClearCase** tab.

The **ClearCase** tab shows the last known names of the versions of ClearCase elements associated with the change request. Files without extended paths have not yet been checked in.

4 Updating a Snapshot View

The rules in the *config spec* of your view are usually designed to select a discrete set of versions from the VOB. For example, your view is usually intended to contain a set of versions that build successfully. However, when other developers check in new versions from their views, a snapshot view may become out of date or inconsistent with the versions in the VOB. To make sure that your view contains the set of versions the config spec selects, you must update it.

This chapter explains

- Starting an update operation
- What happens when you update a view
- Unloading elements

An update operation copies versions of elements from a VOB to your view. Only the checkin operation copies changes from your view back to a VOB.

Starting an Update Operation

You can start an update operation for

- The entire view
- At least one file or at least one directory tree

Updating the Entire View

Update the entire view periodically to make sure you have the correct version of all loaded files and directories.

To Start the Update Tool

1. In the Folder pane in ClearCase Explorer, select the root directory of the snapshot view.
2. Right-click to display the shortcut menu. Then click **Update View**.

3 To change default behavior for the update operation, click **Advanced** in the Start Update dialog box and change the currently selected options. Any changes you make become the defaults for subsequent updates.

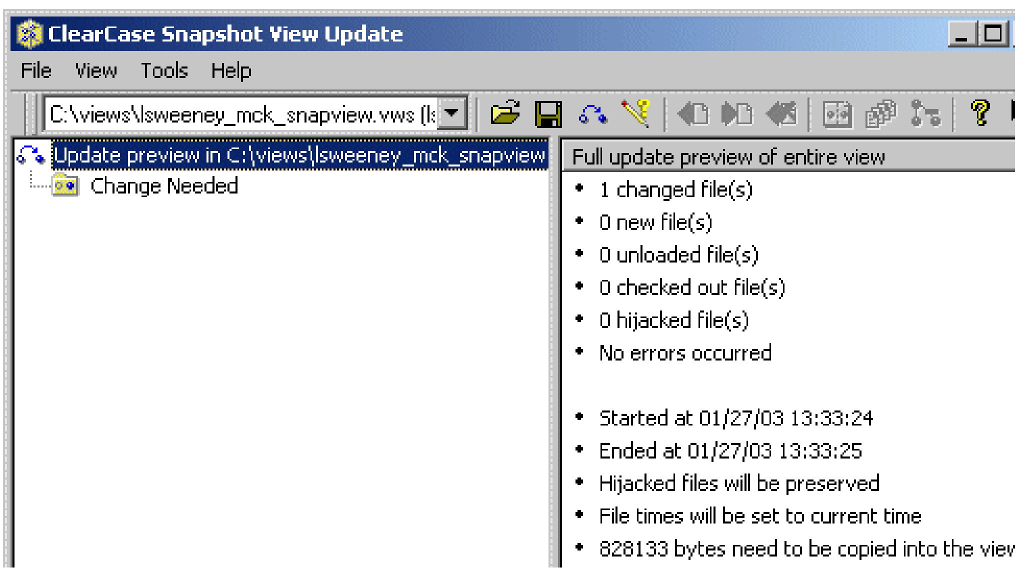
   If you do not change options on the **Advanced** tab, the defaults for the update operation are as follows:

   - For any file or directory that is neither checked out nor hijacked, if the version the config spec selects is different from the version in the view, overwrite it with the version from the VOB.
   - Leave all hijacked files in the view with their current modifications.
   - For any files or directories modified by the update operation, use the time-stamp option specified when the view was first created.

4 Click **OK** to start the update.

   Rational ClearCase begins the update procedure 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 (Figure 17). For a description of this window, see the Help.

Figure 17 The Update Tool Window



Updating Files and Directory Trees

To save time, you can update individual files or directories. (Rational ClearCase updates directories recursively.) Updating only specific parts of your view may eventually cause the view to contain an inconsistent set of versions.

1 In ClearCase Explorer, select the files or directories you want to update.

2 Right-click to display the shortcut menu; then click **Update**.

  ClearCase begins the update procedure and displays a summary of its progress.

Note: You cannot update a checked-out file. To undo changes to a checked-out file and start over with the version in the VOB, cancel the checkout. See *Canceling Checkouts* on page 53.

Tip: To Find a Set of Files

If you know that the files you want to update share a common attribute, such as a similar modification date, you can use the **Search** command in ClearCase Explorer to find the set of files. Then, you can update the files from the Search Results dialog box.

1 In the Folder pane in ClearCase Explorer, right-click and click **Search**. The Search Results dialog box appears.

2 In **Look In**, edit the path to the view or to a specific directory in the view.

3 Use **Search for files or folders named** or **Containing text** to restrict the search.

4 Click **Search Options** and use **Date**, **Type**, **Size**, and **Advanced Options** to specify search criteria.

5 Click **Search Now**.

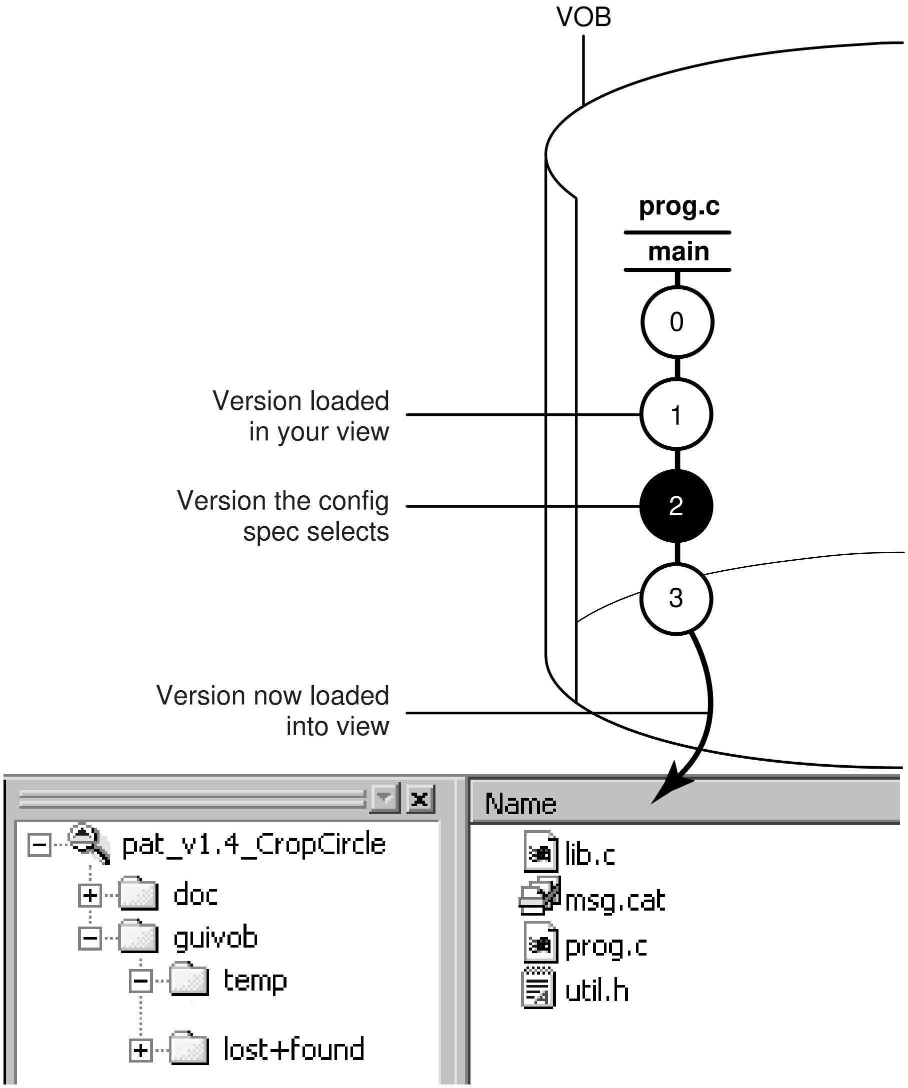
  The Details pane in the Search Results dialog box displays the files and directories found.

6 In the Details pane, select the files to update, right-click on one of the selected files, and click **Update**.

Under the Hood: What Happens When You Update a View

When you start an update operation, ClearCase compares the version of the elements loaded in the view with the version the config spec selects in the VOB. If the config spec selects a version in the VOB that is different from the version loaded in your view, ClearCase copies the version from the VOB into your view (Figure 18). ClearCase does not make this comparison or otherwise modify versions currently checked out to the view.

Figure 18 Update Operation



The update operation takes into account the fact that changes may be occurring in the VOB during the update. As ClearCase updates your view, other developers may check in new versions of elements that the load rules of your view select. To avoid loading an inconsistent set of versions, the update operation ignores versions in the VOB that meet both of the following conditions:

- The version was checked in after the moment the update began.
- The version is now selected by a config spec rule that involves the **LATEST** version label.

The update operation adjusts for the possibility that the system clocks on different hosts in a network may be out of sync (clock skew).

Unloading Elements

If the config spec of a view no longer selects an element, ClearCase removes, or unloads, it from the view. Unloading does not affect view-private files or view-private directories.

Updating can cause an element to be unloaded from a view in the following situations:

- You remove the load rule that specifies the element (or that specifies a directory element somewhere above it). For information about removing load rules, see *To Change Which Elements Are Loaded into a Snapshot View* on page 103.

- The version-selection rules no longer select any version of the element. This can happen when your config spec selects a version of the parent directory that no longer contains a version of the file element.

Unloading Files

The action that ClearCase takes to unload a file depends on the current state of the file:

- For a file that is not checked out, ClearCase deletes the file from the view.

- For a *hijacked* file, ClearCase appends .unloaded to the file name, unless you set the Update Tool to delete hijacked files. (Hijacked files are discussed throughout Appendix A, *Working in a Snapshot View Without the Network*.) You change the settings on the **Advanced** tab in the Start Update dialog box. See *Updating the Entire View* on page 61.

- For a checked-out file, ClearCase appends .unloaded to the file name. The version remains checked out to your view.

Unloading Directories

ClearCase unloads directories recursively. To unload a directory element, ClearCase unloads the files in the directory. If any view-private objects, hijacked files, or checked-out files are in the directory, or if the directory is currently in use (for example, if your current working directory is in or below the directory) ClearCase appends

.unloaded to the name of the directory. For example, if the directory src contains view-private files, ClearCase renames the directory to src.unloaded.

Working On a Team 5

The development cycle presented so far is a fairly simple one in which everyone in an organization contributes to the same development project. But a software development cycle often involves several concurrent development projects. For example:

- You may want to experiment with some changes to the graphic user interface as a result of feedback from usability testing, but are not yet sure whether to include your changes in official builds.
- Another team may try to optimize the database schema without upsetting the current one.
- Another group may need to get a head start on a feature for the next release of the product.

This chapter describes the functions that Rational ClearCase provides to support parallel development. Parallel development is a style of working in which teams use the same set of source files for different, concurrent development projects:

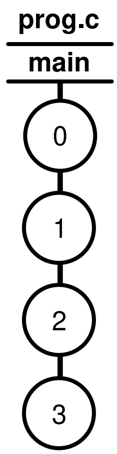
- Version trees
- Working on branches
- Merging
- Sharing control of a branch in an environment by using Rational ClearCase MultiSite

(You do not need to read the section about sharing control of a branch with developers at other sites unless your project manager or MultiSite administrator directs you.)

The Version Tree

Each time you revise and check in an element, ClearCase creates a new version of the element in the VOB. This linear progression is illustrated by Figure 19.

Figure 19 Linear Progression of Versions



ClearCase can organize the different versions of an element in a VOB into a version tree. Like any tree, a version tree has branches. Each branch represents an independent line of development. Changes on one branch do not affect other branches unless you merge. In Figure 20, **main**, **pat_usability**, and **db_optimize** are branches being used to develop different releases of the file element prog.c concurrently.

Under the Hood: The Initial Version on a Subbranch

When you create a subbranch for an element, which is any branch below the **main** branch, the initial version contains the same data as the version from which you start the branch (Figure 21). (The initial version on the **main** branch contains no data. For more information, see *Excluding Elements* on page 104.)

Figure 20   Version Tree of a File Element

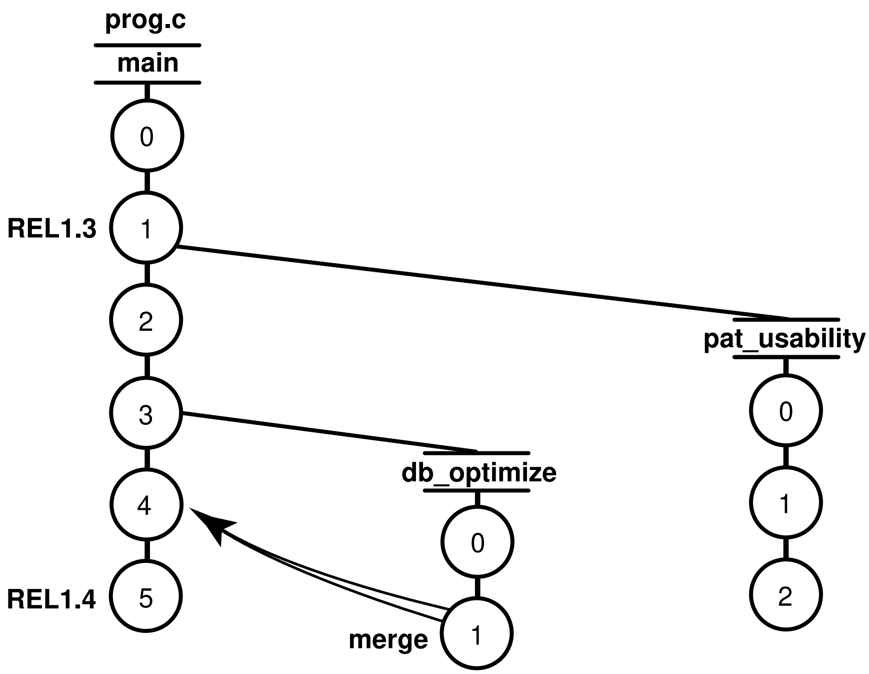
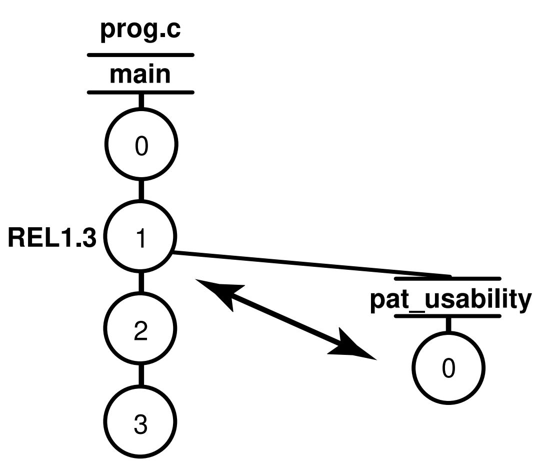


Figure 21   The Initial Version on a Subbranch



Working on Branches

The policies of your organization may dictate that each development project use its own branch to isolate its changes from other development projects. To adhere to this policy, each member of a project team uses a view whose config spec specifies the following information:

- The versions to select in the specific branch of the development project. As Figure 22 illustrates, some or all source files for the project have at least one version on the specified branch. If an element does not have a version on the specified branch, other rules in the config spec select a version of the element. In Figure 22, because lib.c does not have a version on the **pat_usability** branch, the view selects the version on the **main** branch.
- A special *make branch* rule. When this view checks out a version, the make-branch rule creates the branch of the development project (if it does not already exist).

For example, each member of the project team that is optimizing the database schema uses a view that selects versions on the **db_optimize** branch and creates new versions on that branch.

For more information about branches, see *Managing Software Projects* and the **mkbranch** reference page in the *Command Reference*.

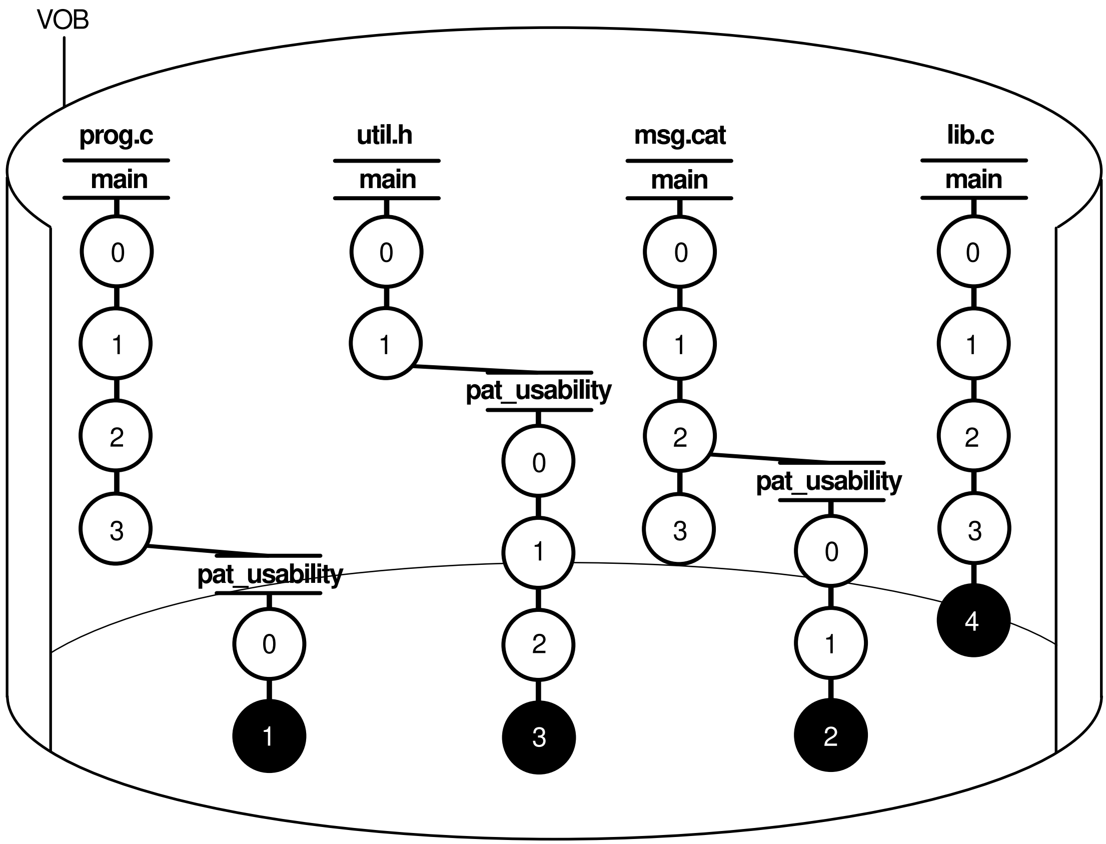
The Version-Extended Path

ClearCase commands and documentation use a notation to specify a version of an element on a branch. For example, util.h@@\main\2 specifies version 2 of util.h on the **main** branch; util.h@@\main\r1_bugs\bug404\1 specifies version 1 of util.h on the **bug404** subbranch below the **r1_bugs** subbranch, which is below the **main** branch (Figure 23).

From a command-line interface, you can use version-extended paths to access versions other than the ones currently selected by your view. To view the contents of a version that is not currently in a snapshot view, you must use the **cleartool get** command in addition to version-extended paths.

For information about the syntax for version-extended paths, see the **pathnames_ccase** reference page in the *Command Reference*.

Figure 22 Elements Have Common Branches



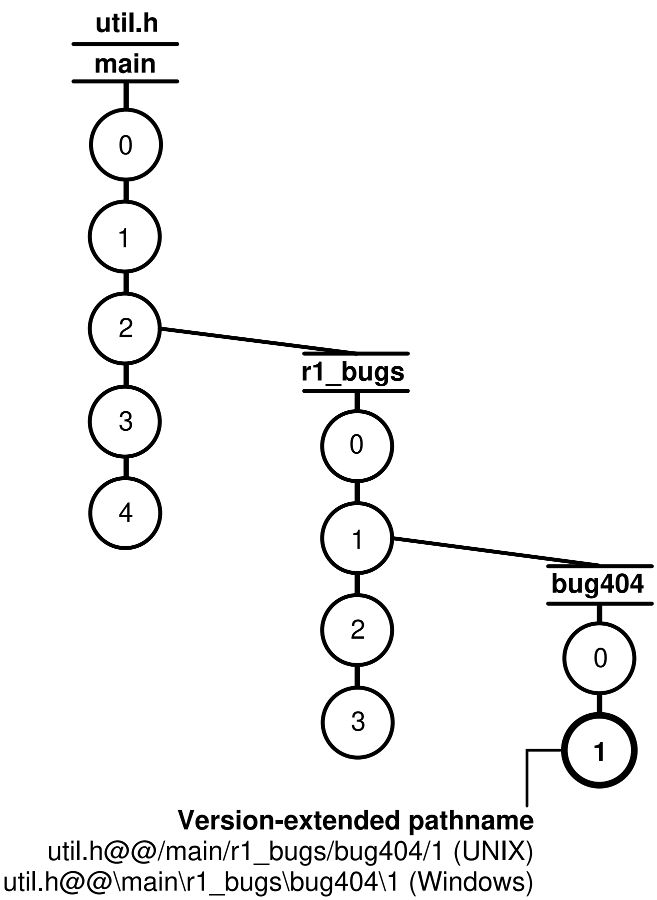
Load Rules

```
load  \guivob\prog.c
load  \guivob\util.h
load  \guivob\msg.cat
load  \guivob\lib.c
```

Version-selection Rules

```
element * \main\pat_usability\LATEST
element * \main\LATEST -mkbranch pat_usability
```

Figure 23 Version-Extended Paths



Merging

In a parallel development environment, the opposite of branching is merging. In the simplest scenario, merging incorporates changes on a subbranch into the **main** branch. However, you can merge work from any branch to any other branch. ClearCase includes automated merge facilities for handling almost any scenario.

One of the most powerful of ClearCase features is source control of directories. Each version of a directory element catalogs a set of file elements and directory elements. In a parallel development environment, directory-level changes may occur as frequently as file-level changes. All the merge scenarios discussed in this chapter apply to both directory and file elements.

This section describes the following merge scenarios:

- Merging all changes made on a single subbranch (see *Scenario: Merging All Changes Made on a Subbranch* on page 79)
- Merging selectively from a single subbranch (see *Scenario: Selective Merge from a Subbranch* on page 80)
- Removing the contributions of some versions on a single subbranch (see *Scenario: Removing the Contributions of Some Versions* on page 83)
- Recording merges that occur outside ClearCase (see *Recording Merges That Occur Outside ClearCase* on page 84)

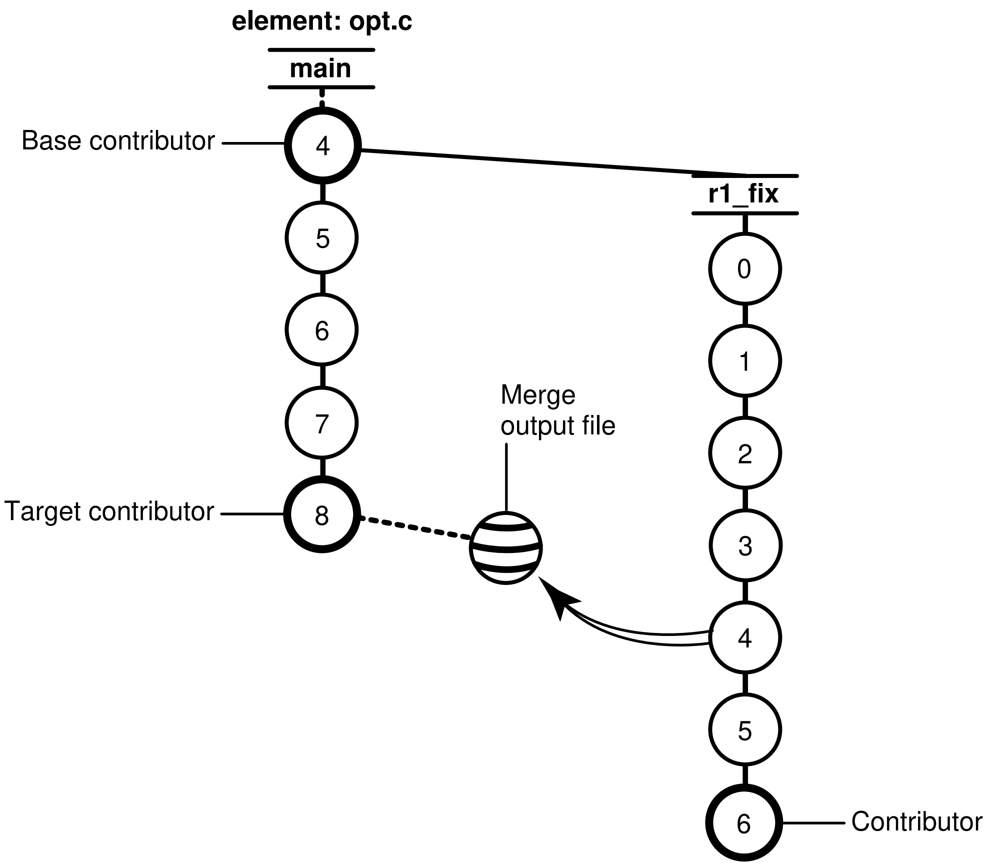
ClearCase also supports merging work from many branches to a single branch; this is typically a project manager's or integrator's task (see *Managing Software Projects*).

Under the Hood: How ClearCase Merges Files and Directories

A merge combines the contents of two or more files or directories into a new file or directory. The merge algorithm uses the following files during a merge (Figure 24):

- 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.) If all the contributors are versions of the same element, ClearCase determines which contributor is the base contributor (but you can specify a different one). For more information about determining a base contributor, see *Determination of the Base Contributor* on page 76.
- The target contributor, which is typically the latest version on the branch that will contain the results of the merge. You determine which contributor is the target contributor.

Figure 24 Versions Involved in a Typical Merge



- The merge output file, 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.

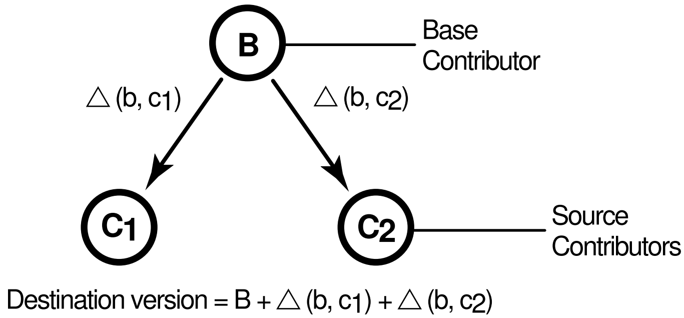
To merge files and directories, ClearCase takes the following steps:

1. It identifies the base contributor.
2. It compares each contributor against the base contributor (Figure 25).
3. It copies any line that is unchanged between the base contributor and any other contributor to the merge output file.
4. For any line that has changed between the base contributor and one other contributor, ClearCase accepts the change in the contributor; 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.

Figure 25 ClearCase Merge Algorithm



File Merge Algorithm

A merge is a straightforward extension of a file comparison. Instead of displaying the differences, Diff Merge analyzes them (sometimes with your help) and copies sections of text to the output file:

- Sections in which there are no differences among the contributors are copied to the output file.
- When *one* contributor differs from the base contributor, Diff Merge accepts the change and copies the modified section of the contributor to the output file:

```
------------[changed 3-4]----|--------[changed to 3-4 file 2]---
now is the thyme                     | now is the time
for all good men                     | for all good people
                                    -|-
*** Automatic: Applying CHANGE from file 2 [lines 3-4]
============
```

(You can turn off automatic acceptance of this kind of change.)

- When two or more contributors differ from the base contributor, Diff Merge detects the conflict, and prompts you to resolve it. It displays all contributor differences and allows you to accept or reject each one.

```
              [changed 10]          |          [changed to 10 file 2]---
cent                                | sent
                                   -|-
              [changed 10]          |          [changed to 10 file 3]---
cent                                | scent
                                    |-
Do you want the CHANGE made in file 2? [yes] no
Do you want the CHANGE made in file 3? [yes] yes
Applying CHANGE from file 3 [line 10]
============
```

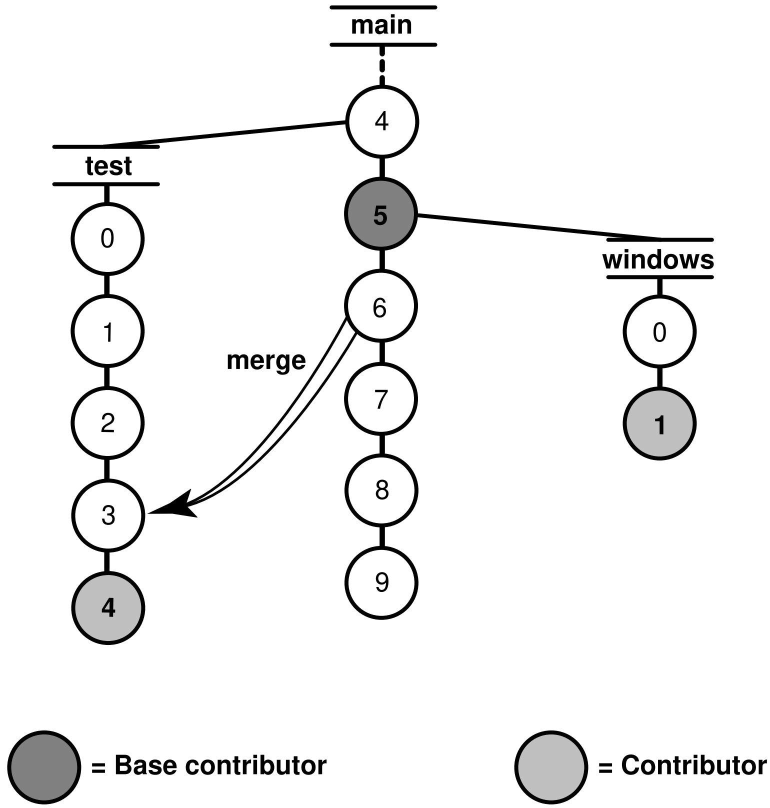
Be sure to verify that the changes you accept produce consistent merged output. For example, after performing a merge involving file util.h, you can compare files util.h.contrib (which contains its previous contents) and the new util.h (which contains the merge output).

Determination of the Base Contributor

If all the contributors are versions of the same element, Diff Merge determines the base contributor automatically. It examines the version tree of the element, which includes all the merge arrows created by previous merge operations. This examination reveals the relationships among versions from the standpoint of their contents (which versions contributed their data to this version?), rather than their creation order (which versions were created before this version?). Diff Merge selects as the base contributor the closest common ancestor in this enhanced version tree.

Figure 26 illustrates a common case of merging. If no merges have been performed in the element, the actual common ancestor (**5**) of the contributors (**test/4** and **windows/1**) in the version tree is selected to be the base contributor.

Figure 26 Determination of the Base Contributor for a Merge



If the contributors are not all versions of the same element, there is no common ancestor (or other base contributor). In this case, ClearCase turns off automated merging, and you must resolve all discrepancies among the contributors.

Recording of Merge Arrows

Under the following conditions, the merge is recorded by creating one or more merge arrows (hyperlinks of type **Merge**):

- All contributor files must be versions of the same file element.
- One of the contributors must be a checked-out version, and you must specify this version as the target to be overwritten with the merge output (the **–to** option in the **merge** command). (Alternatively, you can optionally create merge arrows without performing a merge; in this case, you do not need to check out any of the contributors.)

- You must not perform the merge but suppress creation of merge arrows.
- You must not use any of these options: selective merge, subtractive merge, or base contributor specification (the **–insert**, **–delete**, and **–base** options in the **merge** command).

Diff Merge draws an arrow from each contributor version (except the base contributor) to the target version. You can see merge arrows with the Version Tree Browser.

Locating Versions with Merge Hyperlinks

The **find** and **lsvtree –merge** commands can locate versions with **Merge** hyperlinks. The **describe** command lists all of the hyperlinks of a version, including merge arrows:

cleartool describe util.h@@/main/3

```
version "util.h@@/main/3"
.
.
.
 Hyperlinks:
 Merge@278@/vob_3 /vob_3/src/util.h@@/main/rel2_bugfix/1
 -> /vob_3/src/util.h@@/main/3
```

Directory Merge Algorithm

Each version of a ClearCase directory element contains the names of certain file elements, directory elements, and VOB symbolic links. Diff Merge can process two or more versions of the same directory element, producing a directory version that reflects the contents of all contributors. The algorithm is similar to that for a file merge. Diff Merge prompts for user interaction only when two or more of the contributors are in conflict.

One of the directory versions—the merge target—must be checked out. (Typically, it is the version in your view.) Diff Merge updates the checked-out directory by adding, removing, and changing names of elements and links.

> **Note**: A directory merge does not leave behind a .contrib file, with the pre-merge contents of the target version.

Merging Directories

We recommend that you use this procedure when merging directories:

1. Ensure that all contributor versions of the directory are checked in.
2. Check out the target version of the directory.

3 Perform the directory merge immediately, without making any other changes to the checked-out version.

If you follow this procedure, it easier to determine exactly what the merge accomplished. Enter a **diff –predecessor** command on the checked-out version, which has just been updated by **merge**.

Using ln and rmname in Diff Merge

ClearCase uses VOB hard links to implement directory merges. You can use the **ln** and **rmname** commands to perform full or partial merges manually. See the **ln** and **rmname** reference pages in the *Command Reference*.

Scenario: Merging All Changes Made on a Subbranch

Merging all changes made on a subbranch is the simplest and most common scenario (Figure 27).

Bug fixes for an element named opt.c are being made on branch **r1_fix**, which was created at the baseline version **RLS1.0** (**\main\4**). Now, all the changes made on the subbranch are to be incorporated into **main**, where a few new versions have been created in the meantime.

Task Overview

Merging the changes from the **r1_fix** branch involves the following tasks:

1 Access your dynamic view (see *Starting Dynamic Views* on page 39) or snapshot view (see *Accessing Snapshot Views* on page 40).

   The view must select the target version, which, in Figure 27, is opt.c@@\main\8.

2 If the target version is checked out to your view for other revisions, create a pre-merge checkpoint by checking it in. To make it easier to find this checkpoint, consider labeling the version.

3 Use the Merge Manager to find elements with versions on a specific subbranch and automatically merge any nonconflicting differences. For example, in Figure 27, you find elements with versions on the **r1_fix** subbranch.

   In your project, several elements might have versions on the **r1_fix** branch. With the Merge Manager, you can choose for which elements you merge changes from one branch to another.

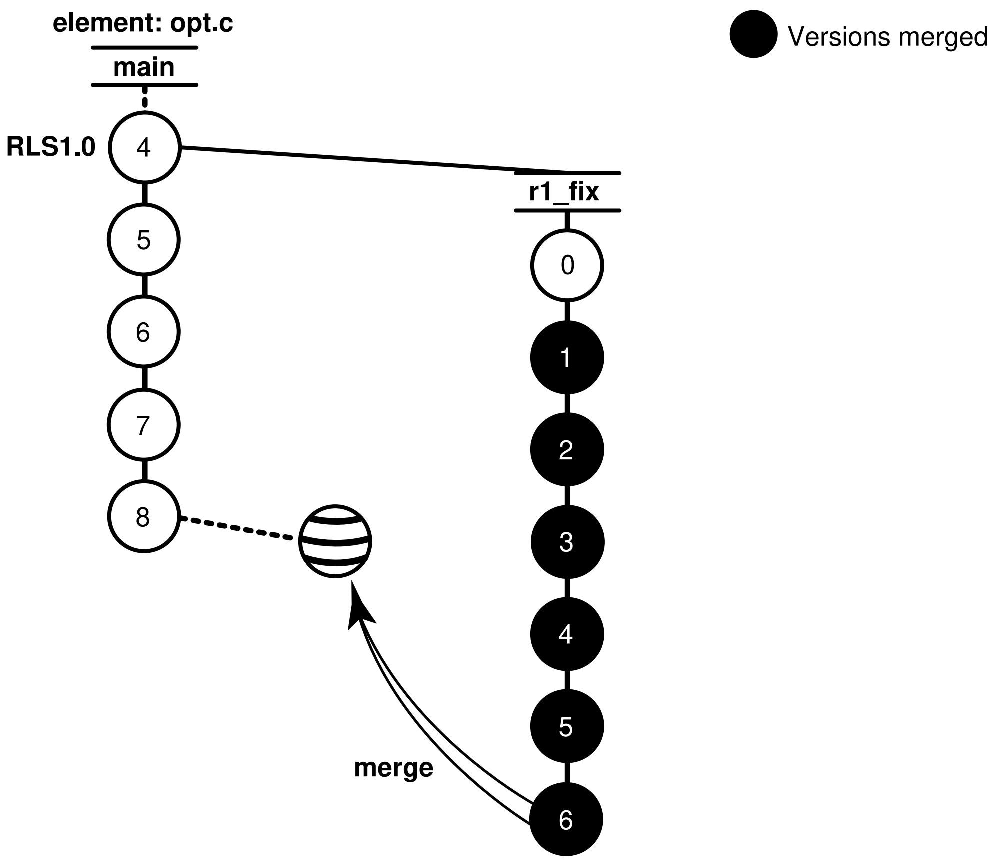
4 Use Diff Merge to resolve any conflicting differences between merge contributors.

5 Test the merge results in the view you start in Step 1. Then check in the target version (which contains the results of the merge).

Getting More Information

For detailed information about completing this task, see ClearCase Help. For information about starting Help, see *To Access Online Information* on page 1.

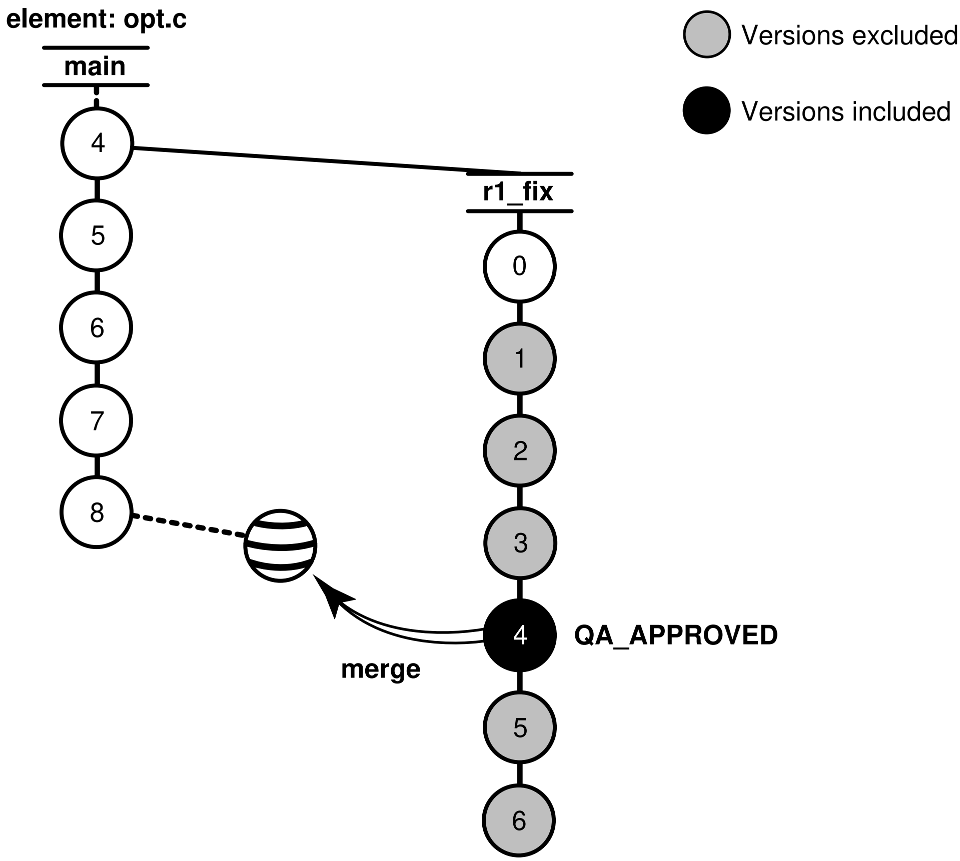
Figure 27 Merging All Changes from a Subbranch



Scenario: Selective Merge from a Subbranch

In the selective merge scenario, the project manager wants to incorporate into new development several lines of code that were added in version **\main\r1_fix\4** (Figure 28).

Figure 28 Selective Merge from a Subbranch



It is critical that you merge only the lines of code as written in this version: it was used and verified to fix a specific bug that prevents further development on the new project.

Selective merges can be tricky: versions you exclude as contributors to the merge may contain needed revisions. For example, if the function you added in **\main\r1_fix\4** relies on a variable definition that was added in **\main\r1_fix\2**, you must include version 2 in the merge.

Merging a Range of Versions

You can also specify a single range of consecutive versions to contribute to the merge. For example, if you need the variable definitions added in **\main\r1_fix\2** as well as the code added in **\main\r1_fix\4**, you can include versions 2 through 4 in the merge.

Task Overview

Merging selective versions from the **r1_fix** branch involves the following tasks:

1. Access your dynamic view (see *Starting Dynamic Views* on page 39) or snapshot view (see *Accessing Snapshot Views* on page 40).

   The view must select the target version, which in Figure 28 is opt.c@@\main\8.

2. If the target version is checked out to your view for other revisions, create a pre-merge checkpoint by checking it in.

3. To determine which versions contain changes that you want to merge to the target version, use the Version Tree Browser and the History Browser. In a snapshot view, use either the **cleartool get** command or **Send To** command in the Version Tree Browser or History Browser to see the contents of versions not loaded into your view. (For information about opening a version not currently in your view, see *Accessing Elements Not Loaded into a Snapshot View* on page 101 and *Adjusting the Scope of a View* on page 102.)

4. To start the merge, check out the target version, and then issue the **cleartool merge** command with the **–insert –graphical** arguments. (You cannot start a selective merge from Diff Merge.)

   For example, the following commands merge only the changes in version 4 on the **r1_fix** branch:

   cleartool checkout opt.c
   cleartool merge –graphical –to opt.c –insert –version \main\4

   These commands merge only the changes in versions 2 through 4 on the **r1_fix** branch:

   cleartool checkout opt.c
   cleartool merge –graphical –to opt.c –insert –version \main\r1_fix\2 \main\4

5. In Diff Merge, complete the merge. Then save the results and exit. For information about using Diff Merge, see the Help.

6. Test the merge results in the view you start in Step 1. Then check in the target version.

   Note: In a selective merge, ClearCase does not create a merge arrow. A merge arrow indicates that all the data of a version has been merged, not parts of it.

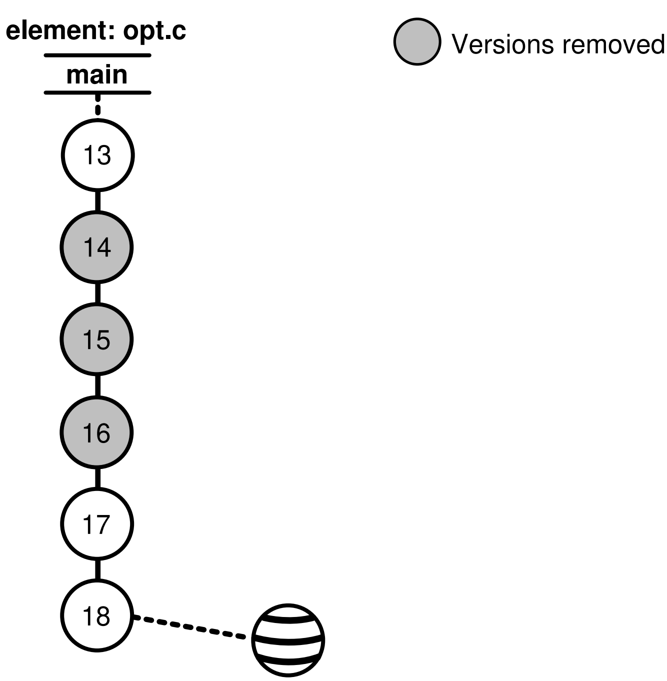
Getting More Information

For detailed information about completing this task, see the **merge** and **version_selector** reference pages in the *Command Reference* or see Help. For information about starting Help, see *To Access Online Information* on page 1.

Scenario: Removing the Contributions of Some Versions

The project manager has decided that a new feature, implemented in versions 14 through 16 on the **main** branch, will not be included in the product. You must perform a subtractive merge to remove the changes made in those versions (Figure 29).

Figure 29 Removing the Contributions of Some Versions



Task Overview

A subtractive merge is the opposite of a selective merge: it removes from the checked-out version the changes made in one or more of its predecessors. Performing a subtractive merge involves the following tasks:

1. Access your dynamic view (see *Starting Dynamic Views* on page 39) or snapshot view (see *Accessing Snapshot Views* on page 40).

The view must select the branch from which you want to remove revisions.

2 If the target version is checked out to your view for other revisions, create a pre-merge checkpoint by checking it in. In Figure 29, the target version is opt.c@@\main\18.

3 To determine which versions contain changes you want to remove, use the Version Tree Browser and the History Browser. From a snapshot view, use either the **cleartool get** command or the **Send To** command in the Version Tree Browser or History Browser to see the contents of versions not loaded into your view. (For information about opening a version not currently in your view, see *Accessing Elements Not Loaded into a Snapshot View* on page 101 and *Adjusting the Scope of a View* on page 102.)

4 To perform the merge, check out the target version, and then use the **cleartool merge** command with the **–delete –graphical** arguments. (You cannot start a subtractive merge from Diff Merge.)

For example, the following commands remove revisions to versions 14 through 16 on the **main** branch:

cleartool checkout opt.c
cleartool merge –graphical –to opt.c –delete –version \main\14 \main\16

5 In Diff Merge, complete the merge. Then save the results and exit. For information about using Diff Merge, see the Help.

6 Test the merge results in your view. Then check in the target version (which contains the results of the merge).

Note: In a subtractive merge, ClearCase does not create a merge arrow. A merge arrow indicates that data has been merged, not removed.

Getting More Information

For detailed information about completing this task, see the **merge** and **version_selector** reference pages in the *Command Reference* or see Help. For information about starting Help, see *To Access Online Information* on page 1.

Recording Merges That Occur Outside ClearCase

You can merge versions of an element manually or with any available analysis and editing tools. To update the version tree of an element with a merge that occurs outside ClearCase, do the following:

1 Check out the target version.

2 Perform the merge with your own tools.

3 Check in the target version.

4 Then record the merge by drawing a merge arrow from the contributors to the new version that contains the result of the merge.

After you draw the merge arrow, your merge is indistinguishable from one performed with ClearCase tools.

For example, use the following commands to merge a version of nextwhat.c on the **enhance** branch to the branch currently selected by your view:

```
cleartool checkout nextwhat.c
Checkout comments for "nextwhat.c":
merge enhance branch
.
Checked out "nextwhat.c" from version "\main\1".
```

<*use your own tools to merge data into checked-out version*>

```
cleartool merge –to nextwhat.c –ndata –version ...\enhance\LATEST
Recorded merge of "nextwhat.c".
```

The **–ndata** option suppresses the merge but creates merge arrows as if ClearCase had merged the versions.

Getting More Information

For detailed information about completing this task, see the **merge** and **version_selector** reference pages in the *Command Reference* or see Help. For information about starting Help, see *To Access Online Information* on page 1.

Sharing Control of a Branch with Developers at Other Sites

Note: This section describes how to request control of a branch from another development site. Do not read this section unless your project manager or MultiSite administrator directs you to.

If your company uses MultiSite to distribute development among multiple geographical sites, you may share source files with developers at other sites. Each site has its own replica of the VOB, and developers work in their site-specific replica (known as the *current replica*). Each replica controls (masters) a particular branch of an element, and only developers at the site of that replica can work on that branch. In this scenario, MultiSite branch mastership does not affect you, and you can do your work as usual.

However, sometimes elements cannot have multiple branches. For example, some file types cannot be merged, so development must occur on a single branch. In this scenario, all developers must work on a single branch (usually, the **main** branch). MultiSite allows only one replica to master a branch at any given time. Therefore, if a developer at another site needs to work on the element, she must request mastership of the branch.

> **Note**: The developer can also request mastership of branch types. For more information, see the *Administrator's Guide* for Rational ClearCase MultiSite.

For example, the file doc_info.doc cannot be merged because it is a file type for which you do not have a *type manager*, but developers at different sites need to make changes to it. If the branch is mastered by your current replica, you can check out the file. If the branch is mastered by another replica, you cannot check out the file. If you try to check out the file, ClearCase presents an error message:

- In the graphical interface:

  ```
  Branch '\main' is not mastered by the current replica. Master
  replica of branch is 'sanfran_hub'. Only unreserved, nonmastered
  checkout is allowed at the current replica.
  ```

- On the command line:

  ```
  cleartool: Error: Cannot checkout branch 'main'.
  The branch is mastered by replica 'sanfran_hub'.
  Current replica is 'boston_hub'.
  cleartool: Error: Unable to check out 'doc_info.doc'.
  ```

For you to check out the file reserved or to check in the file after a nonmastered checkout, your current replica must master the branch. You can request mastership through the graphical interface or with a **cleartool** command.

If you have permission to request mastership from the master replica of the branch, if mastership requests are enabled, and if there are no blocking conditions, then the mastership change is made at the master replica, and a MultiSite update packet that contains the change is sent to your current replica. When your current replica imports the packet, it receives mastership of the branch and you can check out the file.

> **Note**: Authorizing developers to request mastership and enabling mastership requests at a replica are tasks performed by the MultiSite administrator. For more information, see the *Administrator's Guide* for Rational ClearCase MultiSite.

When you use mastership requests to transfer control of a branch, you can use either of two methods to do your work:

- Request mastership of the branch and wait for mastership to be transferred to your current replica; then perform a reserved checkout. You must use this method if you cannot or do not want to merge versions of the element.

- Request mastership of the branch and use a nonmastered checkout to check out the branch immediately. You may have to perform a merge before you can check in your work.

The following sections describe both methods.

To Request Mastership of a Branch and Wait for the Transfer

From the command line:

1. At a command prompt, enter a **cleartool reqmaster** command for the branch you need to check out.

   cleartool reqmaster –c "add info re new operating systems" read_me_first.doc@@\main

   ```
   read_me_first.doc@@\main: Change of mastership at sibling replica
   "sanfran_hub" was successful.
   Mastership is in transit to the new master replica.
   ```

2. Wait for mastership to be transferred to your current replica. To list the master replica of a branch, use **describe**:

   cleartool describe read_me_first.doc@@\main

   ```
   branch "read_me_first.doc@@\main"
     created 15-May-99.13:32:05 by sg.user
     branch type: main
     master replica: boston_hub@\doc
   ...
   ```

   In this example, your current replica is **boston_hub** in the VOB family **\doc**. The output of the **describe** command shows that **boston_hub** is the master replica of the branch, which means that you can check out the branch as reserved.

3. Perform a reserved checkout, edit the file, and check in your work.

For detailed information about requesting mastership from the graphical interface, see Help. You can request mastership from the Find Checkouts window, the Merge Manager, or the Version Tree Browser.

To Check Out the Branch Before Mastership Is Transferred

If you can merge versions of the element you need to check out, you can work on the file while you wait for mastership to be transferred to your replica.

To use this method from the graphical interface:

1. In the Details pane in ClearCase Explorer, right-click the element you want to check out and click **Check Out**.

2 In the Check Out dialog box, select **Unreserved if already reserved**.

3 If the Confirm Version to Check Out dialog box is open, select **Request mastership of the branch** and click **Yes**.

4 In the Request Mastership dialog box, click **Request Mastership**.

5 Make changes to the element.

6 Wait for mastership to be transferred to your current replica. To list the master replica of a branch, use the Property Browser:

   a Right-click the element and click **Version Tree**.

   b In the Version Tree, right-click the branch icon and click **Properties**.

   c Click the **Mastership** tab.

7 Check in the element. If the checkin succeeds, you are finished. If the checkin fails because you have to perform a merge, proceed to Step 8.

8 Use the Merge Manager to merge the latest version on the branch to your checked-out version.

9 Check in the file.

To use this method from the command line:

1 Enter a **reqmaster** command for the branch you need to check out.

   cleartool reqmaster –c "fix bug #28386" prog.c@@\main\integ

```
prog.c@@\main\integ: Change of mastership at sibling replica
"lib_lex" was successful.
Mastership is in transit to the new master replica.
```

2 Use **cleartool checkout –unreserved –nmaster** to perform a nonmastered checkout.

   cleartool checkout –c "fix bug #28386" –unreserved –nmaster prog.c@@\main\integ

3 Make changes to the element.

4 Wait for mastership to be transferred to your current replica. To list the master replica of a branch, use **describe**:

   cleartool describe \lib\prog.c@@\main

```
branch "\lib\prog.c@@\main"
  created 15-May-99.13:32:05 by nlg.user
  branch type: main
  master replica: lib_london@\lib
...
```

5 Check in the element. If the checkin succeeds, you are finished.

cleartool checkin –nc prog.c
```
Checked in "prog.c" version "\main\65".
```

If the checkin fails because you have to perform a merge, proceed to Step 6:

cleartool checkin –nc prog.c
```
cleartool: Error: The most recent version on branch "\main" is not
the predecessor of this version.
cleartool: Error: Unable to check in "prog.c".
```

6 Merge from the latest version on the branch to your checked-out version.

cleartool merge –to prog.c –version \main\LATEST
(if necessary, you are prompted to resolve conflicts)
```
Moved contributor "prog.c" to "prog.c.contrib".
Output of merge is in "prog.c".
Recorded merge of "prog.c".
```

7 Check in the element.

Requesting Mastership After the Checkout

You can perform a nonmastered checkout, but not request mastership at the time of the checkout. You can request mastership at any time by using the **reqmaster** command, or by clicking **Request Mastership** on the shortcut menu for nonmastered checkouts available in Find Checkouts or the Merge Manager.

Setting the Default for Nonmastered Checkouts

You can set the default behavior of the Check Out dialog box so that you always perform a nonmastered checkout if a reserved or unreserved checkout would fail.

1 Do one of the following:

- From ClearCase Explorer, click **Tools > Options** and, under **ClearCase**, click **ClearCase Options**.
- Click **Start > Programs > Rational Software > Rational ClearCase > User Preferences**.

The ClearCase User Options dialog box appears.

2 Click the **Operations** tab.

3 Under **Check Out**, select **Unreserved, nonmastered if branch is mastered by another replica (MultiSite only)**.

When you check out an element in a replicated VOB, the Check Out dialog box opens with **Unreserved, nonmastered** set. When you check out an element in an unreplicated VOB, this setting is ignored.

Troubleshooting

If the request for mastership fails because there are checkouts on the branch at the master replica, try your request again later or ask the other developer to check in the file or directory and then try again. If you receive other errors, contact your project manager or MultiSite administrator.

Other Tasks

6

Chapter 3, *Working in a View*, describes tasks you perform daily or weekly. You may need to perform some of these tasks less often:

- Adding files and directories to source control
- Moving, removing, and renaming elements
- Accessing elements not loaded into a snapshot view
- Adjusting the scope of a view
- Assigning snapshot views to drive letters
- Registering a snapshot view
- Moving views
- Regenerating the view.dat file for a snapshot view
- Accessing views and VOBs across platform types

Adding Files and Directories to Source Control

You can add files or directories to source control at any time. Usually, you add a few files to a directory that is already under source control. Less frequently, you may need to add an entire directory tree to source control.

This section explains these tasks:

- Adding files and directories to an existing directory
- Adding a directory tree for a new development project

To Add Elements to Source Control

Note: The version-selection rules of your view determine the branch on which the first version of an element is created. Make sure the view you use to add elements creates versions on an appropriate *branch*.

To add files and directories to source control from an existing directory tree:

1. In ClearCase Explorer, navigate to the view used for your development task.
2. Navigate to the parent directory to which you want to add the files or directories.
3. If the files or directories are not present, drag them to the parent directory.

4 Select the files and directories you want to add to source control.

5 Right-click one of the selected objects and click **Add to Source Control**.

We recommend that you select items that are the farthest from the root of the directory tree: the **Add to Source Control** command for any given file or directory also adds any parent directories (up to the VOB root directory) that are not already *elements*.

6 Click **OK**.

Under the Hood: What Happens When You Add to Source Control

The **mkelem** or **Add to Source Control** command always creates an element and initializes its version tree by creating a single branch (named **main**) and a single, empty version (version 0) on that branch. The following arguments for the **mkelem** command or options in the Add to Source Control dialog box determine optional ClearCase behavior:

- Setting **Keep checked out** or using **mkelem** with no arguments checks out the element. Any view-private data that corresponds to the element path remains in your view only and is added to version 1 in the VOB when you check in (Figure 30).
- Clearing **Keep checked out** or using **mkelem –ci** checks in the element, using any existing view-private data that corresponds to the element path as the content for version 1. The config spec of your view determines the branch on which ClearCase creates version 1.
- Setting **Preserve file modification time** or using **mkelem -ci -ptime** does not alter the date and time that the file was last modified.
- Clearing **Preserve file modification time** changes the modified time on the file to the checkin time.
- Using **mkelem –nco** suppresses automatic checkout; **mkelem** creates the new element, along with the **main** branch and version **\main\0**, but does not check it out. If *element-path* exists, it is moved aside to a .keep file.
- (Replicated VOBs only) Setting **Make current replica the master of all newly created branches** (file elements only) or using **mkelem –master** or **mkdir –master** assigns to your *current replica* mastership of all branches created during element creation. You will be able to create new versions on the branches.

  Clearing this check box or using **mkelem** or **mkdir** without the **–master** option assigns mastership of a new branch to the VOB replica that masters the associated branch type. If this replica is not your current replica, you cannot create new versions on the branch.

You can set the default for mastership assignment in the User Options dialog box:

a In ClearCase Explorer, click **Tools > Options**.

b In the Options dialog box under **ClearCase**, click **ClearCase Options**.

c In the ClearCase User Options dialog box, click **Operations**.

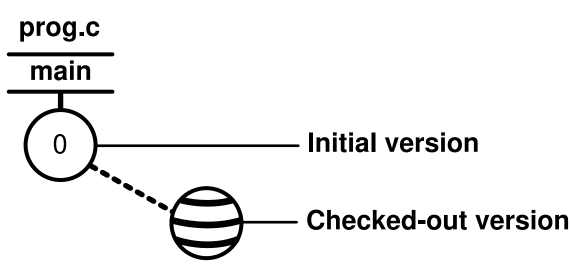
d Under **Check In**/**Add to Source Control**, click **Advanced Options**.

e Select **When creating an element in a replicated VOB, make current replica the master of all newly created branches**.

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

Other views do not see the element until the parent directories of the element are checked in (the **Add to Source Control** command does this for you) and you check in the file or directory.

Figure 30 Creating an Element



Note: *VOB links* make it possible to have more than one copy of a directory in a snapshot view. When you add a file or directory to a snapshot view, ClearCase updates all other instances of the parent directory that are loaded into your view.

File Types and Element Types

Each element is an instance of an element type. You can specify an element type with the **–eltype** option. (The **lstype –kind eltype** command lists the element types in a VOB.) The element type must already exist in the VOB in which you are creating the new element, or must exist in the administrative VOB hierarchy associated with the VOB in which you are creating the new element. A **mkelem –eltype directory** command is equivalent to a **mkdir** command.

If you do not specify an element type on the command line, **mkelem** uses the magic files to determine an element type with which to perform file-typing. This involves matching patterns in the name of the new element (and examining the existing view-private file with that name, if one exists; see *Conversion of View-Private Files to Elements* on page 94). If file-typing fails, an error occurs and no element is created:

```
cleartool: Error: Can't pick element type from rules in ...
```

For more information about file-typing, see **cc.magic** in the *Command Reference*.

Access Mode

Elements are controlled by ClearCase permissions, as described in the **permissions** reference page in the *Command Reference*, not by the access modes for files and directories that are set by the operating system. When your view selects a checked-in version of an element, all of its write permissions are turned off. When you check out an element, write permissions are added to the view-private copy. (For more information, see *Under the Hood: What Happens When You Check Out a File or Directory* on page 45.)

When you convert a view-private file to an element (see *Conversion of View-Private Files to Elements* on page 94), its access mode is changed to read-only. To add execute permission for an executable element, use **protect –chmod +x** (see the **protect** reference page in the *Command Reference*).

Conversion of View-Private Files to Elements

You can use the **mkelem** command to convert a view-private file to a file element with the same name. There are several cases:

- By default, **mkelem** creates an empty version 0 of the new element, then checks out the new element to your view. The view-private file becomes the checked-out version of the new element.
- If you use the **–ci** option (check in), **mkelem** does all of the above, then proceeds to check in version 1 of the new element. Thus, version 1 has the contents of the view-private file. With **–ptime**, **mkelem** preserves the modification time of the file being checked in.
- If you use the **–nco** option (no check out), **mkelem** creates an empty version 0 of the new element.

During the element-creation process, the view-private is renamed to prevent a name collision that would affect other ClearCase tools (for example, triggers on the **mkelem** operation). If this renaming fails, a warning message appears. There are two renaming procedures:

- If a new element is checked out, **mkelem** temporarily renames the view-private file, using a .mkelem (or possibly, .mkelem.*n*) suffix. After the new element is created and checked out, **mkelem** restores the original name. This action produces the intended effect: the data formerly in a view-private file is now accessible through an element with the same name.

- If no checkout is performed on the new element, **mkelem** alerts you that the view-private file has been renamed, using a .keep (or possibly, .keep.*n*) extension. Note that the view-private file is not converted to an element; it is moved out of the way to allow creation of an element in its place.

  Note: If **mkelem** does not complete correctly, your view-private file may be left under the .mkelem file name.

Conversion of Nonshareable Derived Objects to Elements

mkelem does not perform any special processing for a nonshareable derived object. The process is the same as for a shareable derived object, as described in *Conversion of View-Private Files to Elements* on page 94. However, when you check in version 1 of the new element, the checkin converts the nonshareable derived object to a shareable derived object, then checks it in.

> **Note**: When a nonshareable derived object is converted to a shareable derived object, its derived object ID changes. For more information, see *Derived Objects and Configuration Records* in *Building Software*.

Creation of Directory Elements

If you create a new directory element, you cannot use the same name as an existing view-private file or directory, and you cannot use **mkelem** to convert an existing view-private directory structure into directory and file elements. To accomplish this task, use the **clearfsimport** and **clearimport** utilities.

Auto-Make-Branch During Element Creation

If your config spec has a **/main/LATEST** rule with a **–mkbranch** clause, **mkelem** checks out a subbranch instead of the **main** branch. For example, suppose your view has this config spec:

```
element * CHECKEDOUT
element * .../gopher_port/LATEST
element * V1.0.1 -mkbranch gopher_port
element * /main/0 -mkbranch gopher_port
```

The /main/0 rule in the last line allows you to create elements. In this case, a **gopher_port** branch is created for the new element, and this branch is checked out instead of **main**:

cleartool mkelem –c "new element for Gopher porting work" base.h

```
Created element "base.h" (type "text_file").
Created branch "gopher_port" from "base.h" version "\main\0".
Checked out "base.h" from version "\main\gopher_port\0".
```

The **auto-make-branch** facility is not invoked if you use the **–nco** option to suppress checkout of the new element. For more about auto-make-branch, see the **checkout** and **config_spec** reference pages in the *Command Reference*.

Creation of Elements in Replicated VOBs

By default, when you create an element in a replicated VOB, **mkelem** and **mkdir** assign mastership of the main branch of the element to the VOB replica that masters the branch type **main**. If this replica is not your current replica, you cannot create versions on the **main** branch. (You can create versions on other branches if they are mastered by the current replica.)

To assign mastership of the **main** branch of a new element to the current replica, use the **–master** option. The **–master** option also allows auto-make-branch during element creation, even if the branch type specified in your config spec is not mastered by the current replica. In this case, **mkelem** or **mkdir** assigns mastership of newly created branches to the current replica. For example, suppose your view has the following config spec:

```
element * CHECKEDOUT
element * .../gms_dev/LATEST
element * /main/0 –mkbranch gms_dev
```

When you create a new element with **mkelem –master** or **mkdir –master** and do not use the **–nco** option, the command creates the branches **main** and **gms_dev** and assigns their mastership to the current replica.

> **Note**: If you use the **–nco** option with **–master**, only the main branch is mastered by the current replica, because it is the only branch created by **mkelem** or **mkdir**.

Element Object and Version References

You sometimes need to distinguish an element itself from the particular version of the element that is selected by your view. In general:

- Appending the extended naming symbol (by default, @@) to the name of an element references the element itself.
- A simple name (no extended naming symbol) is a reference to the version in the view.

For example, msg.c@@ references an element, whereas msg.c refers to a version of that element. In many contexts (for example, **checkin** and **lsvtree**), you can ignore the distinction. But there are ambiguous contexts in which you need to be careful. For example, you can attach attributes and hyperlinks either to version objects or to element objects. Thus, these two commands are different:

cleartool mkattr BugNum 403 msg.c *(attaches attribute to version)*
cleartool mkattr BugNum 403 msg.c@@ *(attaches attribute to element)*

The first command operates on the version of the element selected in your view, but the second command operates on the element itself.

Caution: Do not create elements whose names end with the extended-naming symbol. ClearCase software cannot handle such elements.

Storage Pools

Physical storage (data containers) for the versions of an element is allocated in the storage pools that **mkelem** assigns. You can change pool assignments with the **chpool** command.

Group Membership Restriction

Each VOB has a group list. If your principal *group* is on this list, you can create an element in that VOB. For more information about group lists, see the **protectvob** reference page in the *Command Reference*.

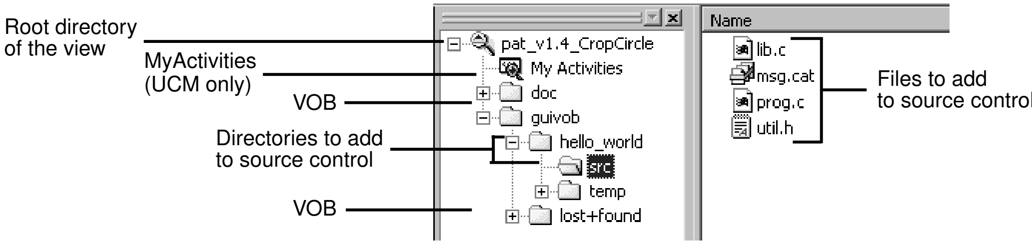
To Add Elements for a New Development Task

Note: This procedure assumes that a VOB exists. For information about creating VOBs, see *Managing Software Projects*.

1. In ClearCase Explorer, navigate to the view you created for the development task.
2. Do **one** of the following, depending on whether you work in a snapshot view or a dynamic view.
   - If you work in a snapshot view, create a folder in the view that corresponds to each VOB to which you want to add files and directories.
   - If you work in a dynamic view, make sure that each VOB to which you want to add files and directories is accessible from your view. If a VOB is not accessible, activate it on your computer (see *To Activate VOBs* on page 40).
3. Under each VOB, do one of the following:
   - Create a directory structure.
   - If files and directories already exist, copy them into the appropriate location within the directory tree.

When you are finished, the view resembles Figure 31.

Figure 31 Directory Structure in a Snapshot View



4 In the Details pane in ClearCase Explorer, select files and directories.

We recommend that you select items that are the farthest from the root of the directory tree: the **Add to Source Control** command for any given file or directory also adds any parent directories (up to the VOB root directory) that are not already *elements*. For example, in Figure 31, if neither \guivob\hello_world nor \guivob\hello_world\src are elements, you can add both directories to source control by adding \guivob\hello_world\src to source control.

5 Right-click one of the selected objects and click **Add to Source Control**.

Importing Files

If you are adding a large number of files and directories to source control, use the **clearfsimport** command (or **clearexport** commands) and **clearimport** command. For more information, see the **clearfsimport** and **clearimport** reference pages in the *Command Reference*.

Moving, Removing, and Renaming Elements

This section explains how to move, remove, and rename elements.

Moving and Removing Elements

Because directories as well as files are under ClearCase control, you can move or remove elements from specific versions of directories without affecting the element itself. Moving or removing elements creates new versions of the parent directories to record the modifications.

For example, version 4 of \gui_vob\hello_world\src contains an element named prog.c. If you remove prog.c from the hello_world\src directory, ClearCase creates version 5 of \gui_vob\hello_world\src, which does not contain the prog.c file element. The element prog.c itself is not modified (Figure 32).

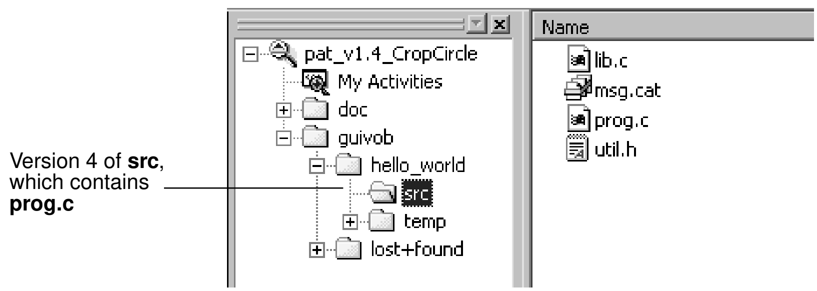
Before you move or remove an element name from a directory, verify with your project manager that your changes will not adversely affect other team members or break project builds.

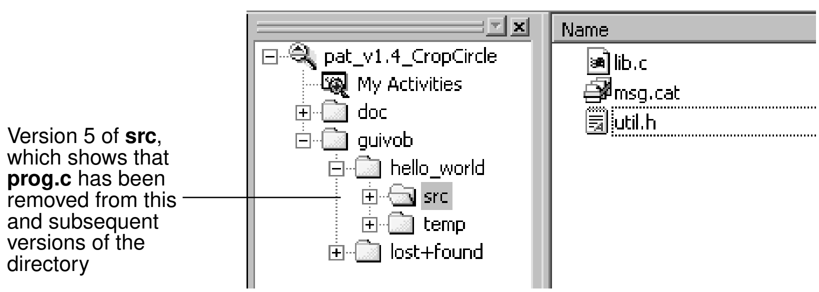
To Move an Element Within a VOB

Do **one** of the following:

- In the Details pane in ClearCase Explorer, drag an element to its destination directory in the Folder pane.
- In the Details pane in ClearCase Explorer, right-click an element and click **Cut**. In the Folder pane, right-click the destination directory and click **Paste**.

Figure 32 Removing an Element Name from a Directory





To Move an Element to Another VOB

Use the **cleartool relocate** command.

Warning: The **relocate** command makes irreversible changes to at least two VOBs and their event histories. We recommend that you not use it for minor adjustments. Furthermore, we recommend that you stop VOB update activity before and during a relocate operation. Check with your project manager and ClearCase administrator before using the **relocate** command.

To Remove an Element Name from a Directory

In the Details pane in ClearCase Explorer, right-click an element and click **Delete**.

Other Methods for Removing Elements

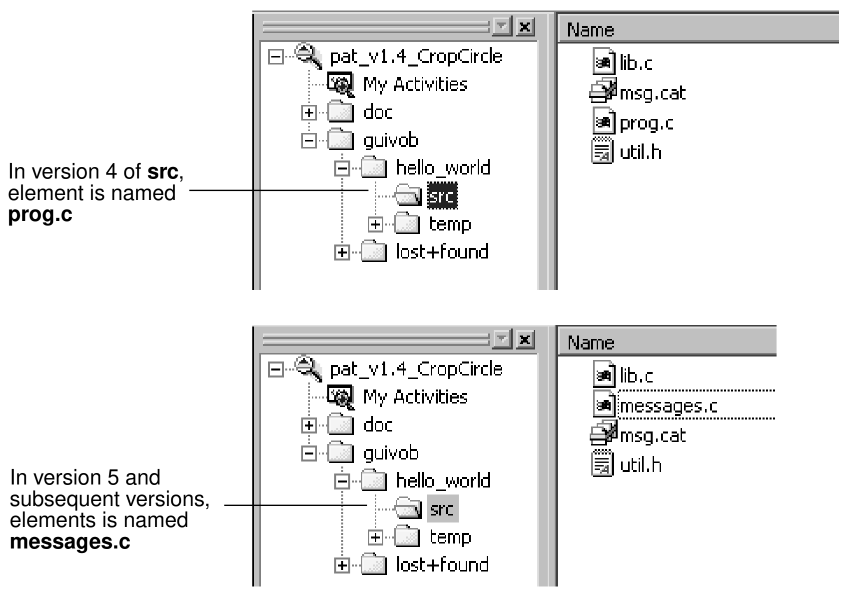
Removing an element name from its parent directory does not affect the element itself, but two other types of a removal operation do irrevocably affect an element, and we recommend that you be very conservative in using these operations:

- Removing a version from the version tree of an element. For more information, see the **rmver** reference page in the *Command Reference*.
- Removing an element from a VOB. For more information, see the **rmelem** reference page in the *Command Reference*.

Renaming Elements

Renaming an element creates a new version of the parent directory to catalog the new element name. The element uses its new name in subsequent versions of its parent directory, but previous versions of the parent directory refer to the element by its previous name (Figure 33).

Figure 33 Renaming an Element



Before you move or remove an element name from a directory, verify with your project manager that your changes will not adversely affect other team members or break project builds.

To Rename an Element

1. In the Details pane in ClearCase Explorer, right-click an element and click **Rename**.
2. In the Details pane, type a new name for the element.
3. Press ENTER.

Accessing Elements Not Loaded into a Snapshot View

While working with source files in a snapshot view, you may need to see the contents of elements that are not loaded into the view or see ClearCase information about these nonloaded elements. For example, you may have chosen not to load a VOB that contains functional-specification documents. However, you may want to check periodically whether the functional specifications have been modified by reviewing the history of the elements.

Listing All Elements in the VOB Namespace

You can set up ClearCase Explorer to provide a directory listing of nonloaded file and directory elements.

To See Nonloaded Elements from ClearCase Explorer

In ClearCase Explorer, click **View** and select **Show Nonloaded Elements**.

(This sets **Show Unloaded Snapshot View Elements** in the Options dialog box under **Snapshot View Options**. Clearing **Show Nonloaded Elements** likewise clears the same option.)

In the Details pane, an icon and the State column indicate which versions are not loaded. ClearCase Explorer displays the version of the element that would be selected if the element were loaded in the view. The Version column in the Details pane indicates which version the view is selecting. Then, you can display the nonloaded element history, version tree, and properties sheet, and compare versions of the element.

To See Metadata for Nonloaded Versions

Metadata is information that ClearCase keeps about elements under source control.

To see metadata for a nonloaded version, click the version in the Details pane and select commands from the **Tools** menu.

Viewing the Contents of a Nonloaded Version

To see the contents of a nonloaded version, you can do either of the following:

- Use the **Send To** command from the shortcut menu of the version in the Version Tree Browser. The **Send To** command uses the set of shortcuts in your Send To folder, which is a function that the Windows operating system provides. For information about your Send To folder, see Windows Help.
- Copy the version into your view with the **cleartool get** command.

You can view nonloaded files or copy them into your view for build purposes, but you cannot check them out. Only file elements that are loaded into the view can be checked out.

Note: You cannot use the **Send To** command or **cleartool get** for directory elements.

To Copy a Nonloaded Version with cleartool get

1. In the Details pane in ClearCase Explorer, in the Version column, note the *version-extended path* of the unloaded element, which is displayed in the Version column.
2. Right-click the parent directory and, from the shortcut menu, click **Command Prompt** to display a **cleartool** prompt.
3. Type the **cleartool get** command and use the following syntax:

   get –to *filename version-extended-path*

For example:

cleartool get –to prog.c.previous.version prog.c@@\main\v3.1_fix\10

This command copies prog.c@@\main\v3.1_fix\10 into your view under the name of prog.c.previous.version. For information on version-extended paths, see *The Version-Extended Path* on page 70.

Adjusting the Scope of a View

At any time during a development cycle, you may need to change the set of files and directories in your view. Two factors determine which files and directories are in your view:

- The set of elements available to your view. In a snapshot view, *load rules* in the config spec determine which elements are available. In a dynamic view, all elements in all VOBs active on your computer are available.
- Within the set of available elements, the *version-selection rules* in the config spec select specific versions.

This section describes the following tasks:

- Changing which elements are loaded into a snapshot view
- Excluding elements
- Activating or deactivating VOBs
- Changing which versions the view selects

To Change Which Elements Are Loaded into a Snapshot View

1. In the ClearCase Explorer Shortcut pane, do one of the following:
   - Click **Toolbox**. Then click **Base ClearCase > Edit View Properties**. In the Edit View Properties dialog box, click **Snapshot**, select the view, and click **OK**.
   - Click **Views**. Then click the page containing the shortcut for the snapshot view and click the shortcut. In the Folder pane, right-click the view root directory and click **Properties of View**.

   The Properties dialog box appears.
2. In the Properties dialog box, click **Load Rules**.
3. On the Load Rules tab, click **Edit load rules**.
4. In the Choose Elements to Load dialog box, do the following.
   - In **Available Elements**, navigate to and select an element. Click **Add** to load another element into the view.
   - In **Selected Elements**, select an entry and click **Remove** to unload a file or directory from the view.

   For information about other commands and options, click **Help**.
5. Click **OK** and **Apply**. The Start Update dialog box appears.

   The update operation makes sure that your view contains the set of versions that the modified config spec selects.
6. You can choose only to load the new files or update the entire view. (Preview is inactive.) We recommend that you clear **Load only from the new load rules** to update the entire view and click **OK**.

7 Update runs and the ClearCase Snapshot View Update dialog box appears. Examine the update results and click **File > Exit** and, in the Properties dialog box, click **OK**.

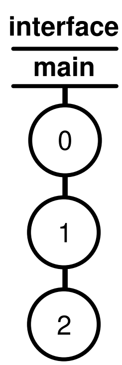
For more information, see *Under the Hood: What Happens When You Update a View* on page 63.

Excluding Elements

ClearCase loads all directory elements recursively. If you want to load only a few elements from a given parent directory, you can add each element separately. For each element you add, a separate *load rule* appears in the *config spec*.

To exclude some elements, you can use an element rule in the *config spec* that selects the initial version of an element on the **main** *branch*. For all ClearCase elements, the initial version on the **main** branch (illustrated in Figure 34) contains no data. (For information about branches, see Chapter 5, *Working On a Team*.)

Figure 34 Initial Version on the Main Branch



For example, to load all elements in the **design** VOB except elements below the interface directory, you can create a load rule that specifies the **design** VOB and an element rule that selects the initial, or empty, version of the interface directory (Figure 35). When you specify the initial version of the interface directory, the entire subtree below it is no longer selected by the config spec and is not loaded.

To Load an Empty Version of an Element

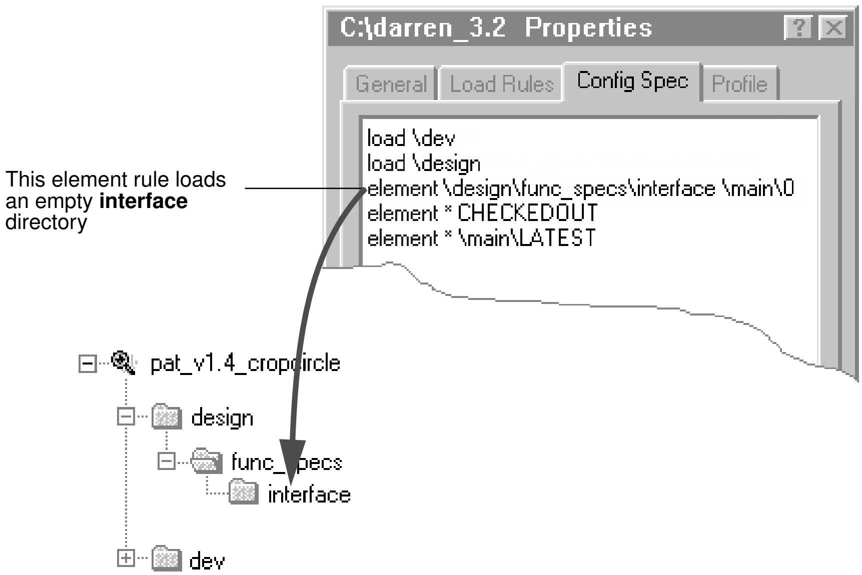
1 In the ClearCase Explorer Shortcut pane, do **one** of the following.

- Click **Toolbox**. Then click **Base ClearCase > Edit View Properties**. In the Edit View Properties dialog box, click **Snapshot**, select the view, and click **OK**.

- Click **Views**. Then click the page containing the shortcut for the snapshot view and click the shortcut. In the Folder pane, right-click the view root directory and click **Properties of View**.

The Properties dialog box appears.

Figure 35 Excluding the interface Directory



2 In the Properties dialog box, click **Config Spec**.

3 On the **Config Spec** tab, click **Edit** to make the rules editable.

4 As illustrated in Figure 35, type an element rule specifying the initial version of the element you want to exclude by using the following syntax:

element *path* **\main\0**

The path must start from the VOB directory.

5 In the Properties dialog box, click **OK**.

Activating or Deactivating VOBs in Dynamic Views

Activating a VOB on your computer makes its files and directories available to your dynamic views (see *To Activate VOBs* on page 40). Deactivating a VOB frees resources on your computer.

To Deactivate VOBs

1. In the system task bar, click **Start > Programs > Rational Software > Rational ClearCase > Administration > Unmount VOB**.
2. In the Unmount dialog box, select one or more VOBs.
3. Click **Unmount**.

To Change the Versions the View Selects

Before completing these steps, see *Working on Branches* on page 70. Depending on the development policies in your organization, the version-selection rules of your view may select versions on a specific branch.

1. In the ClearCase Explorer Shortcut pane, do either of the following.
   - Click **Toolbox**. Then click **Base ClearCase > Edit View Properties**. In the Edit View Properties dialog box, click the appropriate tab, select the view, and click **OK**.
   - Click **Views**. Then click the page containing the shortcut for the view and click the shortcut. In the Folder pane, right-click the root directory of the view and click **Properties of View**.

   The Properties dialog box appears.
2. In the Properties dialog box, click **Config Spec** and then click **Edit**. Modify the version-selection rules either by creating your own rules or using the Windows clipboard to paste a set of rules into the config spec.
3. Click **Apply**.

   For a dynamic view, the selection changes occur immediately.

   For a snapshot view, the Start Update dialog box appears. Click **OK**.

To learn more about *version-selection rules*, see *Managing Software Projects* or the **config_spec** reference page in the *Command Reference*.

Assigning Snapshot Views to Drives

If your makefiles or other files require absolute paths with a specific drive, assign your view to a drive.

Depending on the configuration of your computer, two methods are available:

- Make the snapshot view a shared directory and then assign it to a drive.
- Use the **subst** command.

Assign Only the Root Directory. You must assign only the snapshot view root directory to a drive. The root directory of the snapshot view contains a hidden file named view.dat. ClearCase searches for this file to determine whether the current directory is in a snapshot view.

If you assign a directory below the root of the view to a drive, ClearCase cannot find the view.dat file for the view and assumes that the current directory is not a snapshot view.

Making a Directory Shareable and Assigning a Drive

From ClearCase hosts running either Windows NT or Windows 2000, you can make a directory shareable (see *To Enable Sharing on the Snapshot View Root Directory*) and assign it to a drive (see *To Assign the Snapshot View Shared Directory to a Drive*). With this method, you can access a snapshot view through either My Network Places or Network Neighborhood, but the performance is slightly slower than if you use the **subst** command to assign the view to a drive (see *Using the subst Command for Snapshot View Access*).

From ClearCase hosts running Windows 98, you can assign a shared directory to a drive only if the directory is on some other host; for example, you can use this method to assign someone else's view to a drive.

To Enable Sharing on the Snapshot View Root Directory

Note: If you want to share a directory on a Windows 95 or Windows 98 computer (for example, so team members can assign your view to a drive on their computers), you must enable file sharing by using this procedure.

1. In Windows Explorer, click the snapshot view root directory; then click **File > Sharing**.
2. On the Sharing tab, click **Shared As**.
3. For the sake of consistency, accept the default value in the **Share Name** box. (Make the share name the same as the leaf name of the snapshot view root directory.)
4. Click **OK**.

Other team members can now assign a drive on their computers to this snapshot view root directory (see *To Assign the Snapshot View Shared Directory to a Drive*) or can create a shortcut to the view in ClearCase Explorer (see *To Access Another's Snapshot View from ClearCase Explorer* on page 109).

To Assign the Snapshot View Shared Directory to a Drive

If the root directory of the snapshot view is shareable (see *To Enable Sharing on the Snapshot View Root Directory*), you can assign the directory to a drive on your computer.

> **Note**: If you prefer to use the command line, you can use the **net use** command instead of the following procedure.

1 From Windows Explorer, click **Tools > Map Network Drive**.

2 In the Map Network Drive dialog box, select a letter from the **Drive** box.

3 In **Folder**, enter the path to the shared folder of the snapshot view root directory or click **Browse** and navigate to the root directory of the snapshot view in My Network Places or Network Neighborhood.

4 Click **Finish**.

Using the subst Command for Snapshot View Access

Assigning a snapshot view root directory to a drive letter with the **subst** command provides slightly better performance than making the snapshot view a shared directory (see *Making a Directory Shareable and Assigning a Drive* on page 107). However, only shared directories are accessible through My Network Places or Network Neighborhood. (To assign your own view to a drive on Windows 98 computers, you must use the **subst** command.)

1 Open a command shell.

2 Enter **subst** *driveletter*: *view-path*. For example, the command

   subst Y: C:\library\pat_v1.4_cropcircle

   maps the **pat_v1.4_cropcircle** snapshot view root directory to drive Y.

Assign only the snapshot view root directory to a drive letter. For more information about the **subst** command, type **help subst** in a command shell.

Registering a Snapshot View

If you need to perform ClearCase operations in a snapshot view that you did not create or have not updated, you must register that view.

When you create or update a snapshot view or click on a view shortcut in ClearCase Explorer, ClearCase registers the snapshot view in your computer registry. The ClearCase-Windows Explorer integration running in ClearCase Explorer and Windows Explorer recognizes as ClearCase objects files and directories below a registered snapshot view.

If a snapshot view is not registered on your computer and you access files and directories under the view root directory through Network Neighborhood, the ClearCase-Windows Explorer integration does not recognize the files or directories as ClearCase objects. For example, if you use Windows Explorer to navigate to a team member's snapshot view on a different computer and right-click a file in the view, ClearCase options are not available. Also, if you try to perform ClearCase operations by using an unregistered snapshot view, ClearCase cannot find the view.

To perform ClearCase operations on the files and directories in a snapshot view that is not registered by you (for example, a team member's snapshot view), you must register the snapshot view. You can use ClearCase Explorer (see *To Access Another's Snapshot View from ClearCase Explorer* on page 109) or Windows Explorer (see *To Register a Snapshot View by Using Windows Explorer* on page 109) to register the snapshot view.

To Access Another's Snapshot View from ClearCase Explorer

To access a snapshot view that that is not registered by you, create a view shortcut in ClearCase Explorer. If the root directory of the snapshot view is shareable (see *To Enable Sharing on the Snapshot View Root Directory* on page 107), follow these steps on your computer:

1. Click the **Views** tab, right-click in the Shortcut pane, and click **Add View Shortcut**.
2. In the Add a New View Shortcut dialog box, click the Browse button to the right of the **Snapshot Location** text box; navigate to and select the shareable root directory under My Network Places or Network Neighborhood; and click **OK**.
3. Click **OK**.
4. Click the view shortcut.

This registers the view for your use on any Windows computer if you use a roaming user profile. If you use a local user profile, this registers the view for your use on this one computer. Click the view shortcut to access the snapshot view.

Note: You cannot register a snapshot view that was created from a UNIX host (see *Accessing Views Across Platforms of Different Types* on page 113).

To Register a Snapshot View by Using Windows Explorer

If you want to use Windows Explorer to perform ClearCase operations on the files in a snapshot view that is not registered by you, register the view. If the root directory of the snapshot view is shareable (see *To Enable Sharing on the Snapshot View Root Directory* on page 107), proceed as follows:

1. In Windows Explorer, navigate to the root directory of the view you want to register.

2 Right-click the root directory of the view.

This registers the view for your use on any Windows computer if you use a roaming user profile. If you use a local user profile, this registers the view for your use on this one computer.

Note: You cannot register a snapshot view that was created from a UNIX host (see *Accessing Views Across Platforms of Different Types* on page 113).

Moving Views

This section discusses the following tasks:

- Changing the physical location of the directory tree of a snapshot view
- Moving a view storage directory

For information about changing a view tag, see the **mktag** reference page in the *Command Reference*.

Changing the Physical Location of a Snapshot View

If the snapshot view storage directory is in a storage location, you can use standard Windows commands (such as **Cut** and **Paste** commands or drag and drop operations in Windows Explorer) to move the directory tree of loaded elements and view-private files of the snapshot view.

To Find the Location of the View Storage Directory

1 In the Folder pane in ClearCase Explorer, right-click the view root directory and click **Properties of View**.

2 In the Properties dialog box, click **Advanced**.

The **Host Path** box displays the path for the view storage directory.

Caution: If the view storage directory is located below the root directory of the view (colocated with the view), **do not use** standard Windows commands to move the snapshot view. Instead, see *Moving a View Storage Directory* on page 111.

If the snapshot view storage directory is in a storage location, you can move the view to a different computer, but the computer must run a Windows operating system.

Update After Moving

After moving a snapshot view, you must refresh any view shortcuts in ClearCase Explorer that use the old path to the view. If you did not create a ClearCase Explorer shortcut for the view and you move a view to a different computer, you must register its new location (see *Registering a Snapshot View* on page 108). Until you register its new location, the ClearCase user interface continues to present the old location.

Moving a View Storage Directory

Each dynamic view and snapshot view includes a view storage directory, which ClearCase uses to maintain the view. **Do not use** standard Windows commands (such as **Cut** and **Paste** commands or drag and drop operations) to move a view storage directory for the following reasons:

- The view storage directory includes a database. Moving the database without first shutting down the *view_server* process for the view can corrupt the database.
- ClearCase stores the location of view storage directories in its own set of registries. The information in these registries must be correct for you to perform ClearCase operations in your views. In a dynamic view, the location in ClearCase registries must be correct for you to access any file or directory in the view.

We suggest that you ask your ClearCase administrator to move view storage directories because it may affect other, potentially many other, ClearCase users at your site. For more information about the procedure for moving view storage directories, see the *Administrator's Guide* for Rational ClearCase.

> **Caution**: You will lose data (including view-private files in a dynamic view) if you move a view storage directory without following the procedure described in the *Administrator's Guide* for Rational ClearCase.

Regenerating a Snapshot View view.dat File

The root directory of a snapshot view contains a hidden file, view.dat. If you delete this file inadvertently, ClearCase no longer identifies the view as a ClearCase object, and you can no longer perform ClearCase operations on files or directories loaded in the view.

To Regenerate the view.dat File

1. Open a command shell.
2. Type this command:

   ccperl *ccase-home-dir***\etc\utils\regen_view_dot_dat.pl**
   ^ [**–tag** *snapshot-view-tag*] *snapshot-view-path*

   For example:

```
ccperl   c:\rational\etc\utils  regen_view_dot_dat.pl
^ –tag  pat_v1.4_cropcircle
^ c:\pat_v1.4_cropcircle
```

If the view storage directory is under the root directory of the view, you do not need to use the **–tag** *snapshot-view-tag* argument.

Accessing Views and VOBs Across Platform Types

ClearCase supports environments in which some ClearCase hosts use a Microsoft Windows operating system and others use a UNIX operating system.

This section discusses the following topics:

- Creating views across platform types
- Accessing VOBs across platform types
- Developing software across platform types

Creating Views Across Platforms of Different Types

Your ClearCase administrator can set up storage locations on Windows and UNIX server hosts. Any snapshot view that you create can use one of these storage locations, regardless of the platform type of the server host. For more information about storage locations, see the **mkstgloc** reference page in the *Command Reference*.

For a dynamic view, the view storage directory must be located on a host of the same platform type as the host from which you create the view. If you create a dynamic view from a UNIX host, you must locate the view storage directory on a ClearCase host on UNIX; if you create a dynamic view from a Windows host, you must locate the view storage directory on a Windows NT host that is set up to store view storage directories. We recommend that you locate dynamic view storage directories on the host from which you most often use the view.

Snapshot View Characteristics and Operating-System Type

For snapshot views, the operating system type from which you create the view determines view characteristics; the operating system type that hosts the files and processes related to a snapshot view do not affect the behavior of the view.

For example, it is possible to create a snapshot view from a Windows host and locate the view directory tree and the view storage directory on a ClearCase host on UNIX (assuming that you use third-party software to access UNIX file systems from Windows computers). Although all files related to the view are on a UNIX workstation, because you created the view from a Windows host, the view behaves as if its files are located on a Windows computer:

- Case-sensitive file lookups are not performed in the view.
- The view does not create symbolic links if the load rules encounter a VOB symbolic link.
- You can issue ClearCase commands for the view only from Windows hosts. (ClearCase hosts on UNIX do not recognize the directory tree as a snapshot view.)

Accessing Views Across Platforms of Different Types

This section describes support for accessing a view residing on a platform that differs from the platform from which it is being accessed.

Accessing UNIX Snapshot Views from Windows Hosts

ClearCase supports a set of third-party products for accessing UNIX file systems from Windows computers. If your organization uses one of these products, you can access UNIX snapshot views from Windows Explorer (or a command prompt) just as you would access any other directory tree on a UNIX workstation.

You can access snapshot views across platforms, but you cannot issue ClearCase commands across platforms. For example, you cannot check out files in UNIX snapshot views from Windows hosts nor can you create shortcuts to UNIX snapshot views from ClearCase Explorer.

If, from a Windows host, you hijack a file in a UNIX snapshot view, ClearCase detects the hijack when you update the view from a ClearCase host on UNIX.

Accessing Windows Snapshot Views from UNIX Hosts

ClearCase does not support accessing Windows file systems from UNIX workstations.

Accessing UNIX Dynamic Views from Windows Hosts

ClearCase supports a set of third-party products for accessing UNIX file systems from Windows computers. If your organization uses one of these products, you can complete the following tasks to access UNIX dynamic views from Windows computers:

1. Create the UNIX view with the proper text mode. For more information, see *Developing Software Across Platforms of Different Types* on page 114.
2. Import the view tag of the UNIX view into your Windows network region.

3 Start the dynamic view or add a shortcut to the view in ClearCase Explorer.

Accessing Windows Dynamic Views from UNIX Hosts

ClearCase does not support products for accessing Windows file systems from UNIX workstations. You cannot access Windows views from UNIX hosts.

Accessing VOBs Across Platforms of Different Types

Your ClearCase administrator sets up VOBs on Windows or UNIX hosts and creates *VOB tags* in each ClearCase network region that needs to access the VOBs. (For information about registering UNIX VOB tags in a Windows network region, see the *Administrator's Guide* for Rational ClearCase.) Then, from any ClearCase host on Windows or UNIX systems, you can create snapshot views to load elements from VOBs that have tags in your network region.

From a ClearCase host on Windows that supports dynamic views, you can access VOBs on Windows and UNIX from dynamic views as well as snapshot views. To access VOBs on UNIX from Windows dynamic views, you must use third-party software that provides access to UNIX file systems from Windows computers. From a ClearCase host on UNIX, you cannot access VOBs on Windows from dynamic views. Table 2 summarizes your options for accessing VOBs across platform types.

Table 2 Accessing ClearCase VOBs Across Platform Types

Platform of your ClearCase host	Platform on which VOB is located	View from which you can access source files
Windows computer	Windows computer or UNIX workstation	Snapshot view or dynamic view
UNIX workstation	Windows computer	Snapshot view
UNIX workstation	UNIX workstation	Snapshot view or dynamic view

Developing Software Across Platforms of Different Types

If developers check in source files from views created on both Windows and UNIX hosts, consider creating your views in interop (MS-DOS) text mode. The text modes change how a view manages line terminator sequences. For more information about view text modes, see the *Administrator's Guide* for Rational ClearCase or ClearCase Help. For information about starting Help, see *To Access Online Information* on page 1.

Working in a Snapshot View Without the Network

A

If you need to work with your source files from a computer that is disconnected from the network of Rational ClearCase hosts and servers, you can set up a snapshot view for disconnected use.

This chapter describes the following tasks:

- Setting up a view for your hardware configuration
- Preparing the view
- Disconnecting the view
- Working in the view
- Reconnecting to the network
- Using the Update Tool

Note: While disconnected from the network, you cannot access ClearCase information about the files in your view or issue most ClearCase commands. If you want to work from a remote location and continue to access ClearCase information and issue ClearCase commands, consider using the Web interface for ClearCase. Ask your ClearCase administrator whether the Web interface for ClearCase has been configured at your site and what URL you need to supply to your Web browser to access it. For information about using the Web interface, see the Web interface Help.

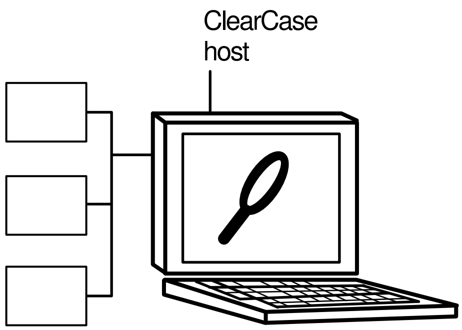
Setting Up a View for Your Hardware Configuration

You can use one of several hardware configurations to work in a snapshot view that is disconnected from the network.

This chapter describes the following recommended configurations:

- Creating and using the view on a laptop computer that periodically connects to the network (Figure 36).

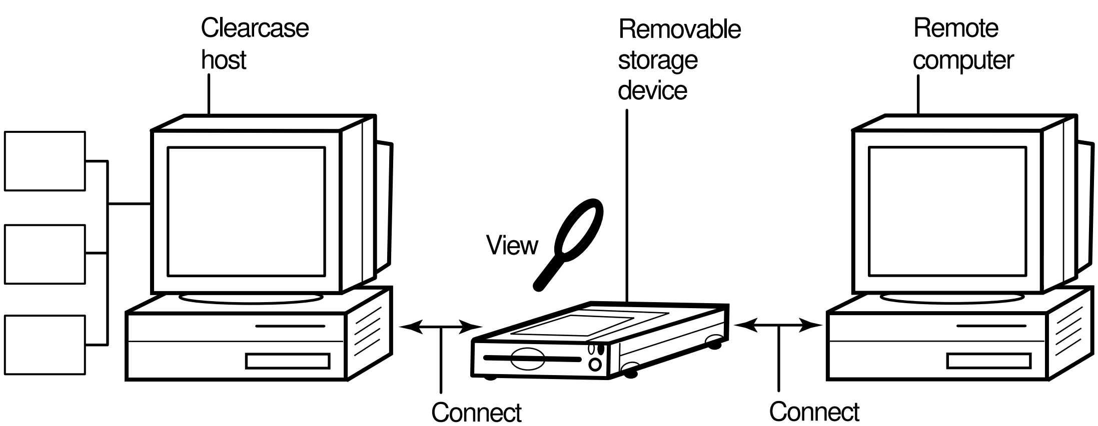
Figure 36 View on a Laptop



Note: The laptop computer must run a Windows operating system.

- Creating and using the view on a removable storage device such as an external hard drive or some other device (such as a Jaz drive) that provides satisfactory read/write performance (Figure 37).

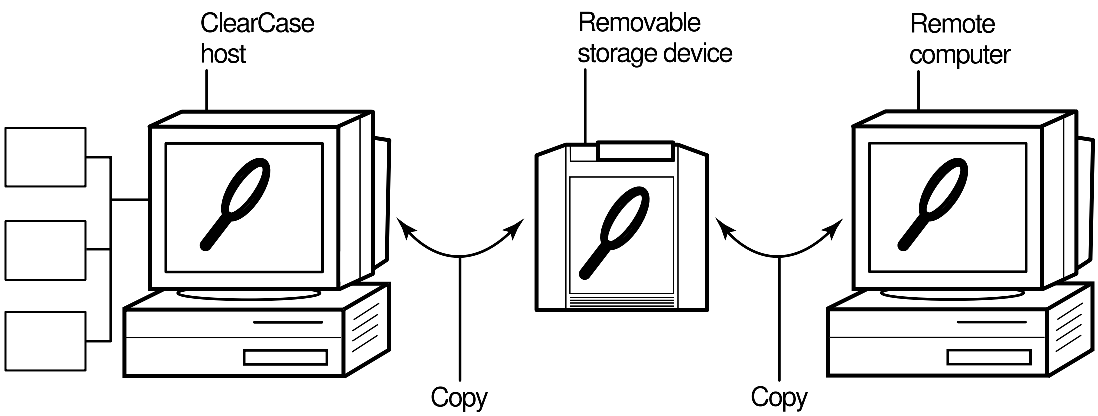
Figure 37 View On a Removable Storage Device



Note: The remote computer must run a Windows operating system.

- Copying the view from a ClearCase host to a temporary, removable storage device such as a diskette or a tape drive, which usually does not provide satisfactory read/write performance, and then copying the view from the storage device to a computer that is disconnected from the network (Figure 38).

Figure 38 Copy the View



Note: The remote computer must run a Windows operating system.

Under the Hood: View Storage in Disconnected-Use Configurations

In all the configurations recommended for disconnected use, the snapshot view storage directory is in a server storage location. We recommend (and in the case of a removable storage device, we enforce) this configuration because the *view_server* process of a view runs on the host that contains the view storage directory or on a host that can access a location on a supported Network Attached Storage (NAS) device which contains the view storage directory.

A view_server is a long-lived process that manages activity for a specific view. If the view storage directory is in the root directory of the snapshot view and you disconnect the view from the network while the view_server process is writing to the storage directory, you can corrupt the data ClearCase uses to maintain your view.

Preparing the View

Before you disconnect the view from the network, complete these tasks:

- Update the view to establish a checkpoint. (For information about updating the view, see *Updating the View* on page 124.)
- Check out the files you expect to modify. After you are disconnected from the network, you cannot check out files, although there are workarounds. (See *Hijacking a File* on page 120.)

When you are no longer connected to the network, you cannot use most ClearCase commands. At this point, the disconnected computer does not distinguish a snapshot view directory from any other directory in the file system.

Disconnecting the View

How you disconnect the view depends on which hardware configuration you use. This section describes disconnecting the view for the following hardware configurations:

- View on a laptop
- View on a removable storage device
- View Copied to a storage device

Hardware Configuration: View on a Laptop

If the view is located on a laptop, you must deactivate the ClearCase-Windows Explorer integration and other developer tools before you disconnect the view. Most ClearCase operations communicate with various ClearCase servers. Even if you install view server software on your host, almost any time you start a ClearCase operation (even an operation as simple as displaying the ClearCase shortcut menu in Windows Explorer), your host must connect to the license server for ClearCase. The ClearCase integration with Microsoft Visual Studio attempts to connect to the license server periodically. If you are disconnected from the network, the application that attempts to connect to the license server for ClearCase locks for several seconds until it returns an error message. Although the application does not fail, such interruptions are unnecessary.

To Deactivate ClearCase Integrations

1. In Control Panel, double-click **ClearCase**.
2. Click the **Options** tab and clear **Connected to network for ClearCase operations**.

3 Disconnect the laptop from the network.

Deactivating the integrations removes ClearCase commands from the ClearCase Explorer and Windows Explorer shortcut menus.

Hardware Configuration: View on a Removable Storage Device

If the view is located on a removable storage device, disconnect the removable storage device from the network computer and reconnect the media to the remote computer.

Hardware Configuration: View Copied to a Storage Device

If you do not have a storage device with satisfactory read/write performance, use the **xcopy** command to copy files from your view to the storage media and from the storage media to the remote computer as follows:

xcopy *path\snapshot-view-directory destination* [**/D** [**:***m-d-y*] | **/M**] **/S /K /R**

For example, **xcopy c:\Library\pat_v1.4_cropcircle e:\temp /D /S /K /R**

The command options accomplish the following:

- The **/D** option instructs **xcopy** to copy only the files that have changed, which is useful if you frequently alternate between the network computer and the nonnetwork computer. Use **/D** if you configured the Update Tool to set the time stamp of updated files to the time at which the files are loaded into the view. You can view or change the Update Tool time stamp option on the **Advanced** tab of the Start Update dialog box. For more information, see Step 3 of *Updating the View* on page 124.

  If you set the Update Tool to create time stamps based on the version-creation time, consider using the **/M** option, which causes **xcopy** to copy the files with the archive attribute set. The **/M** option is less reliable than the **/D** option, because some applications alter the archive attribute without writing the file.

- The **/S** option instructs **xcopy** to copy all subdirectories of the snapshot view.
- The **/K** option keeps the read-only file attribute as established by ClearCase.
- The **/R** option allows **xcopy** to overwrite read-only files.

Working in the View

You cannot use most ClearCase commands when disconnected from the network. Yet you may need to work on files that you did not check out or locate files you have modified. This section provides workarounds for these ClearCase operations.

Hijacking a File

If you need to modify a loaded file element that you have not checked out, you can *hijack* the file. ClearCase considers a file hijacked when you modify it without checking it out. For more information, see *Under the Hood: Determining Whether a File Is Hijacked* on page 124.

When you reconnect to the network, you use the Update Tool to find the files you hijacked. You can do the following with a hijacked file:

- Check out the file. You can then continue to modify it and, when you are ready, check in your changes.
- Undo the hijack. For more information, see *To Undo a Hijack* on page 123.

To Hijack a File

1. In Windows Explorer, right-click the file to hijack and click **Properties** to display the property sheet of the file.
2. On the **General** tab, clear **Read-only**.

To Find Modified Files While Disconnected

You can use the Search applet from either ClearCase Explorer or from **Start > Search > For Files or Folders** to find all files that have been modified after a specified date.

1. Do either of the following:
   - In ClearCase Explorer, right-click on a directory and click **Search**. The directory path is seeded in the **Look in** box.
   - Click **Start > Search > For Files or Folders**.

   The Search Results window appears.
2. In the Search pane, specify the path to the view or to a specific directory in the view.
3. Click **Search Options** and **Date** to define your search criteria for files that have been modified.
4. Click **Search Now**.

   The Details pane expands to display the files and directories it finds.

Reconnecting to the Network

When you return to the office and can access the network directly, do one of the following, depending on your hardware configuration.

Hardware Configuration: View on a Laptop

If you use the view on a laptop, connect the laptop to the network. If ClearCase is installed on the laptop, activate the ClearCase integrations as follows:

1. Click **Start > Settings > Control Panel** and double-click **ClearCase**.
2. Click **Options** and select **Connected to network for ClearCase operations**.
3. Click **OK**.

Hardware Configuration: View on a Removable Storage Device

If you use the view on a removable storage device, connect the removable storage device to the computer on the network.

Hardware Configuration: View Copied to a Storage Device

If you copied the view onto removable media, use **xcopy** with the [**/D** | **/M**] **/S /K /R** options to copy files back to the original location on the network computer.

Using the Update Tool

When you are connected to the network, use the Update Tool for the following tasks:

- Determine how to handle hijacked files
- Update the view

Determining How to Handle Hijacked Files

Handling hijacked files involves the following tasks:

- Finding hijacked files
- Comparing a hijacked file to the version in the VOB
- Checking out a hijacked file
- Merging changes to a hijacked file
- Undoing a hijack
- Choosing other ways to handle hijacked files

To Find Hijacked Files

1 In ClearCase Explorer, right-click the root directory of the snapshot view.

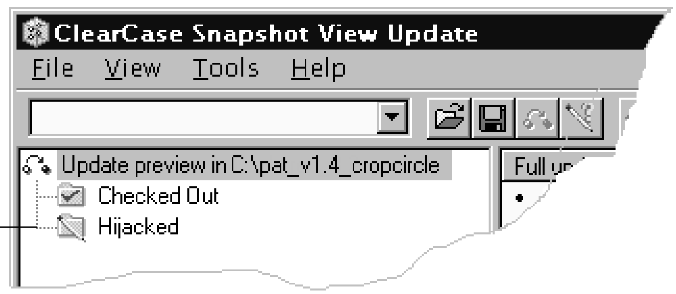
2 On the shortcut menu, click **Find Modified Files**.

Select **Display results when closed** and quit the Modified File Search Complete window.

3 If any hijacked files are in your view, the Snapshot View Update window displays a folder in the left pane titled **Hijacked** (Figure 39). Select **No** for the option asking whether you want to check out the hijacked files now.

Figure 39 Hijacked Files in the Results Window

Hijacked folder



To Compare a Hijacked File to the Version in the VOB

You can use the Diff Merge tool to see how the hijacked file differs from the checked-in version of the file:

1 In the right pane of the Snapshot View Update window, right-click a hijacked file.

2 On the shortcut menu, click **Compare with Original Version**. For information about using the Diff Merge tool, see the Help.

To Check Out a Hijacked File

To keep the modifications in a hijacked file, check out the file:

1 In the right pane of the Snapshot View Update window, right-click a hijacked file.

2 On the shortcut menu, click **Check Out**.

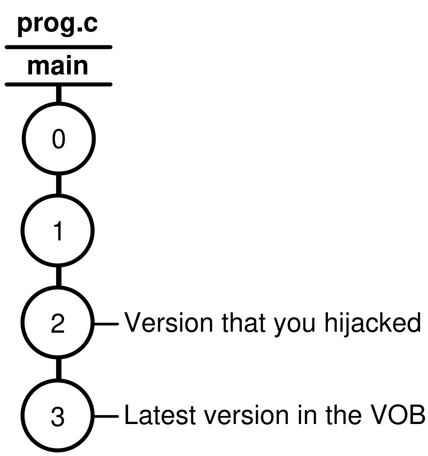
3 ClearCase treats a checked-out hijacked file as it does any other checkout.

When you are ready, you can check in the file.

Merging the Latest Version to a Hijacked File

If you are working with a shared set of versions and someone has checked in a newer version of the file while it was hijacked in your view (Figure 40), ClearCase prevents you from checking in the file.

Figure 40 Hijacked Version May Not Be the Latest Version



You have to merge the latest version in the VOB with the checked-out file before ClearCase allows the checkin.

When you issue the **Check In** command for a nonlatest version, ClearCase opens the dialog box to ask whether you want to merge the file now. If you choose to merge, ClearCase attempts to merge automatically, starting the Diff Merge tool if it needs your input to complete the merge. For information about using Diff Merge, see Help. After the merge, ClearCase prompts you to check in the file.

To Undo a Hijack

If, for specific hijacked files, you want to discard your changes and get a fresh copy of the version from the VOB, you can undo the hijack.

1 In the right pane of the Snapshot View Update window, select one or more hijacked files.

2 Right-click the selected files, and click **Undo Hijacked File**.

ClearCase overwrites the hijacked file with the version that was loaded in the view. If you want to overwrite hijacked files with the versions the config spec selects in the VOB, see Step 3 in *Updating the View* on page 124.

Under the Hood: Determining Whether a File Is Hijacked

To keep track of file modifications in a snapshot view, ClearCase stores the size and last-modified time stamp of a loaded file (as reported by the Windows file system). ClearCase updates these values each time you check out a file, check in a file, or load a new version into the view.

To determine whether a file is hijacked, ClearCase compares the current size and last-modified time stamp of a non-checked-out file with the size and time stamp recorded in the view database. If either value is different from the value in the view database, ClearCase considers the file hijacked.

Changing the read-only attribute of a non-checked-out file does not necessarily mean ClearCase considers the file hijacked.

Other Ways to Handle Hijacked Files

While updating the view, you can handle hijacked files in any of the following ways:

- Leave hijacked files in place
- Rename the hijacked files and load the version from the VOB
- Overwrite hijacked files with the version the config selects in the VOB

For more information, see *Updating the View*.

Updating the View

1. In ClearCase Explorer, select the snapshot view root directory.
2. Right-click to display the shortcut menu and click **Update View**.
3. To configure the Update Tool for handling hijacked files, in the Start Update dialog box click the **Advanced** tab and select a method for handling the remaining hijacked files. You have these choices:
   - Leave hijacked files in place
   - Rename the hijacked files and load the selected version from the VOB
   - Delete hijacked files and load the selected version from the VOB
4. To start the update, click **OK**.

Index

D

V