Working in Base ClearCase

Contents

Working in Base ClearCase

Figures

1 ClearCase Concepts

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

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

Recommend Reading Paths

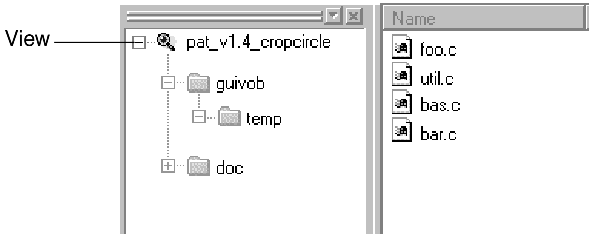
Read this chapter first. Then, if you want to start working immediately, use online help to learn as you go. Or, if you prefer a more structured approach, use the remainder of *Working in Base ClearCase* as a guide through your organization's development cycle.

The sections titled *Under the Hood* provide detailed information and suggest ways to become an advanced ClearCase user.

1.1 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:

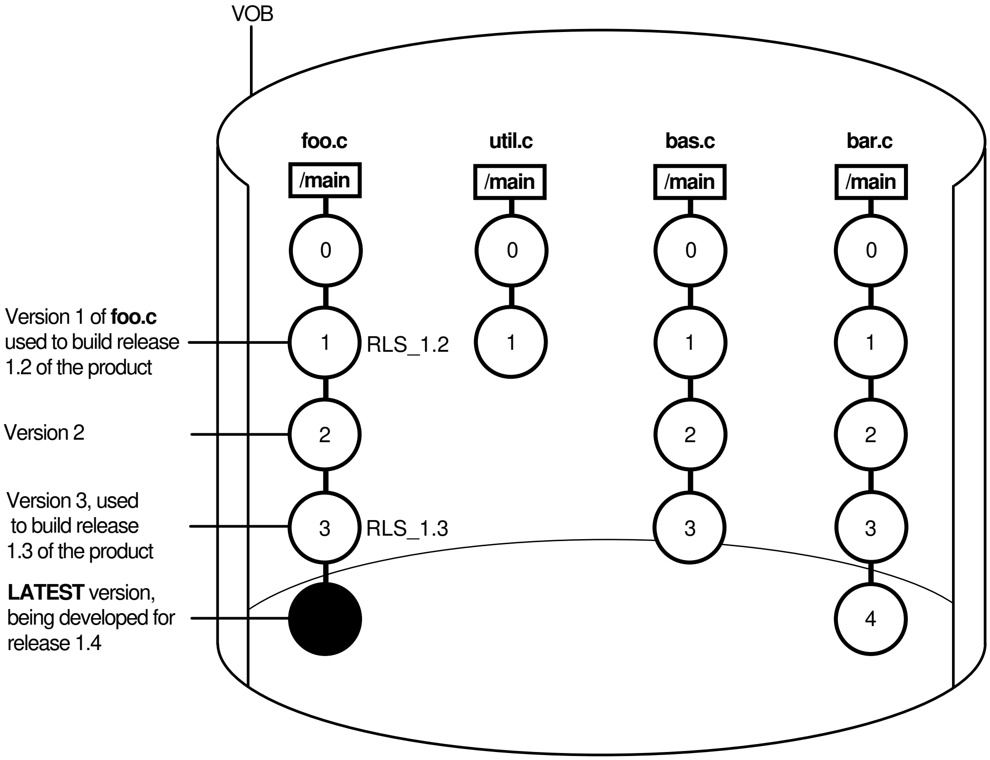
➤ *Snapshot views*, which copy files from data repositories called *VOBs* (*versioned object bases*) to your computer.

➤ *Dynamic views*, which use the ClearCase *multiversion file system* (MVFS) to provide immediate, transparent access to the data in VOBs. (Dynamic views may not be available on all platforms. For more information, see the ClearCase online help.)

1.2 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 **foo.c**, **util.c**, **bas.c**, and **bar.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.

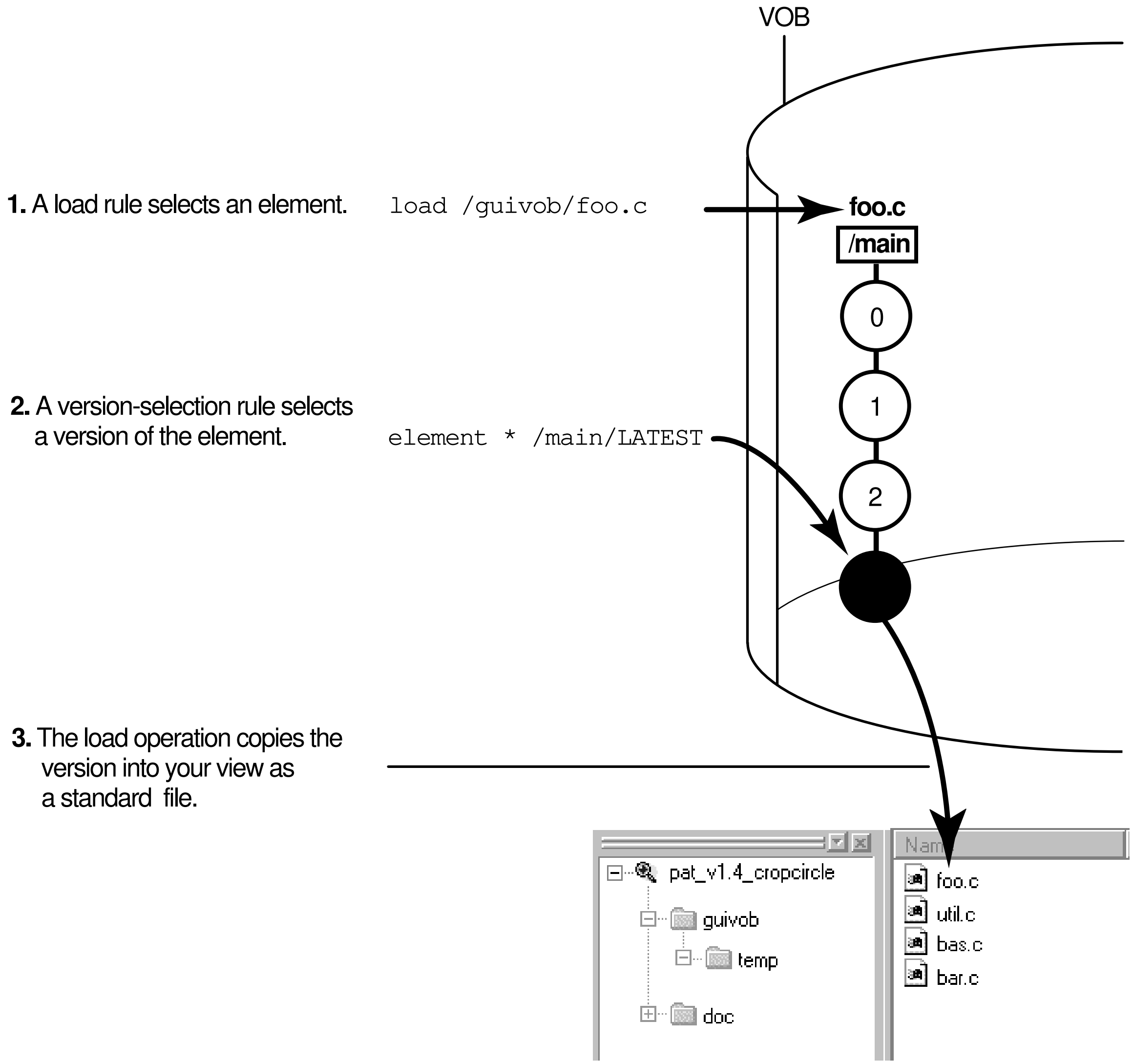
Selecting Elements and Versions

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.

Figure 3 Config Spec Selecting Versions



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. For more information on labels, see *Managing Software Projects with ClearCase*.
- 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.

Learning the Config Spec Syntax

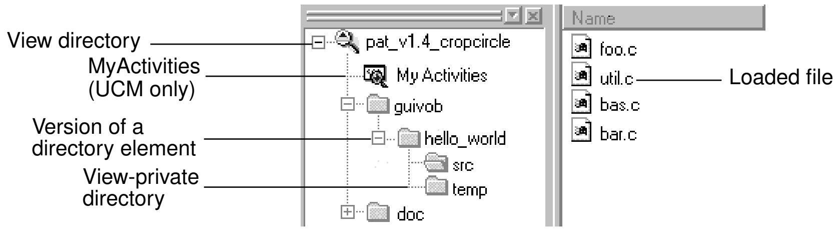
Usually only one or two members of your software team learn the syntax for these rules and create config specs for everyone on the project to use. For more information on the rules in the config spec, see *Managing Software Projects with ClearCase* and the **config_spec** reference page in *ClearCase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

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. 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



1.3 Parallel Development

The combination of config spec rules, views, VOBs, and *branches* (which are 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're working on release 1.4 of a software product, and you want to experiment with the GUI 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 you create from this view evolve separately from the versions used in the official builds. When you're 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.

2 Setting Up a View

Usually you set up 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 92.

The View Creation Wizard assists you in each step of setting up a view. Start the View Creation Wizard and use this chapter to complete the steps.

To start the View Creation Wizard:

1. Start ClearCase Explorer by clicking the shortcut on your desktop or, if you did not install the shortcut, by clicking **Start>Programs>ClearCase>ClearCase Explorer**.

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

3. The first step of the wizard asks if 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*.

2.1 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 the ClearCase online help.)

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 doesn't use the ClearCase *build auditing* and *build avoidance* features.

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 **view** reference page in *ClearCase Reference Manual*.

2.2 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 Location

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

➤ The view's 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.

➤ If you want to access the view from other workstations, it must be located in a shared directory.

➤ 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 your makefiles or other files require absolute pathnames with a specific drive letter, assign the view to a drive letter. For more information, see *Assigning Snapshot Views to Drive Letters* on page 87.

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, and by default ClearCase locates snapshot view storage directories in the storage location. If your ClearCase administrator sets up more than one storage location, ClearCase selects one of these locations as the default when you create a view.

If your ClearCase host is set up to store view storage directories (which happens when you install ClearCase), you can override the default and locate the view storage directory under the root directory of the snapshot view. If you choose to locate 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's **view_server** process is running.

- Make sure the view storage directory is accessible to any data backup schemes your organization institutes.

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

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 Location for a Snapshot View Storage Directory

1. While creating a view, on the step of the wizard that asks you to choose a location for a view, click **Advanced Options**.

2. In the **Advanced Options** dialog box, do one of the following:

   - Select **Use Server Storage Location**. If your administrator created more than one location, ClearCase selects one for you. You can choose a different one if you prefer.

   - 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, select **Co-locate Under Snapshot View Root**. Do not select this option if you plan to use the view while disconnected from the network.

Dynamic View: Choosing a 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 view's owner or contents; 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

A dynamic view's name 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 Letter

If your makefiles or other files require absolute pathnames, assign your view to a drive letter. When you use a wizard to create a view, ClearCase prompts you to assign the dynamic view to a drive letter. After creating a view, if you want to change a drive-letter assignment or assign a drive letter 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-letter assignments from Windows Explorer. Any changes you make to a view's drive-letter assignments outside ClearCase Explorer will invalidate the view's shortcut in ClearCase Explorer.

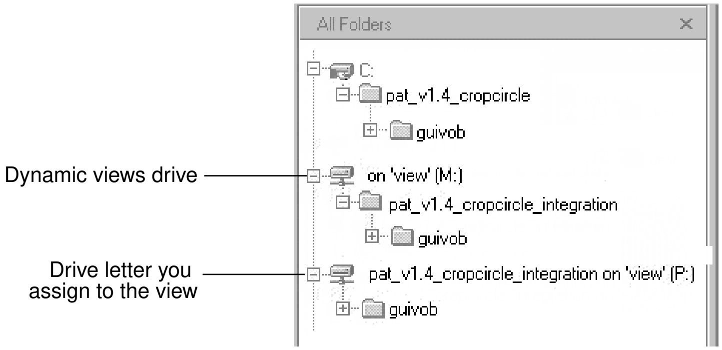
Pathname Differences

You can access any dynamic view that is started on your computer from the *dynamic-views-drive* (**M:** by default) in Windows Explorer. However, when you access a view from the dynamic-views drive, its pathname includes one more component than when you access a view from an assigned drive letter (Figure 5).

For example, both of the following pathnames are available to an element in the **pat_v1.4_cropcircle dynamic** view that is assigned to the **P:** drive:

```
M:\pat_v1.4_cropcircle\guivob\foo.c
P:\guivob\foo.c
```

Figure 5 Dynamic Views Started on a Host



Dynamic View Storage Directories

If this is the first time you're 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, refer to the **clearmake**, **omake,** and **view** reference pages in *ClearCase Reference Manual*.

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 on Windows NT. View processes (specifically, **view_server** processes) run on the machine that physically stores the view storage directory, and only ClearCase hosts on Windows NT can run view processes. The pathname for the directory must not use a Windows special share name. Special share names usually include the dollar sign ($), such as the driveletter$ 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 pathname.

- 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 location for a view, click **Advanced Options**.

2. In the **Advanced Options** dialog box, do one of the following:

   ➢ Select **Use Explicit Path** and provide a UNC pathname 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 administrator created more than one location, ClearCase selects one for you. You can choose a different one if you prefer.

2.3 Adding Version-Selection Rules

Development projects often require team members to add specific version-selection rules to your view's config spec. 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 on creating version-selection rules, see *Managing Software Projects with ClearCase*.

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. Verify with 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*. Verify with 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 one of the following:

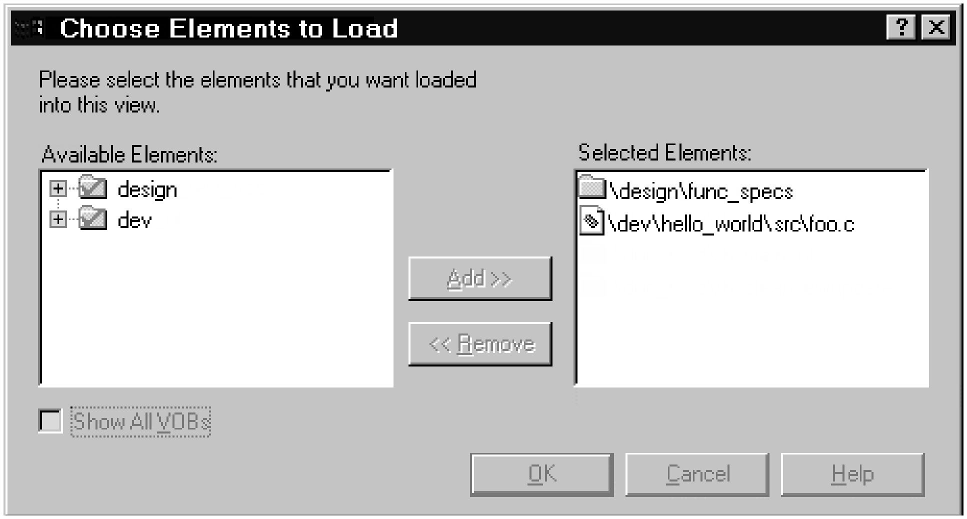
   ➢ For snapshot views, take note of the path for the view's root directory. This is the directory from which you access source files and directories. Then click **OK** to select elements to load into the view.

   ➢ For dynamic views, take note of any drive letter you assigned. Then click **OK**. By default, ClearCase starts the view.

2.4 Snapshot View: Selecting Elements to Load

After you set up version-selection rules for a snapshot view, the View Creation Wizard starts the VOB Namespace Browser (Figure 6) from which you select the files and directories to load into the view.

Figure 6 Choosing Elements to Load



To Choose Elements

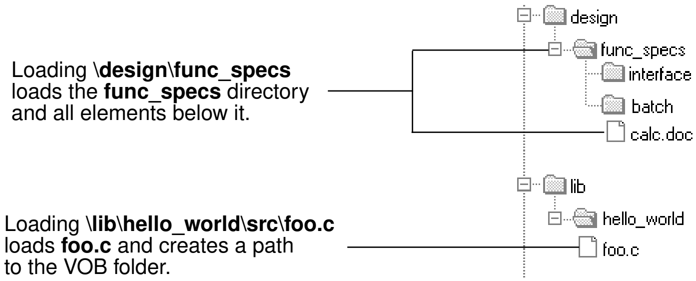
To choose the elements you want to load into your view, browse through the **Available Elements** list. To load an *element*, move it to the **Selected Entries** list. Keep the following in mind:

➤ 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 file's path, ClearCase also copies into the view the empty directories leading up to the VOB directory.

Figure 7 illustrates the results of the load operation.

Figure 7 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 6 creates these two load rules:

```
load \design\func_specs
load \dev\hello_world\foo.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\foo.c**. To maintain the path to **foo.c**, the View Creation Wizard creates the directory **\dev\hello_world\src** in the view.

Case-Sensitivity

If you're selecting elements from VOBs located on a UNIX host, you may encounter problems 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 While Disconnected from the Network*.)

Administering ClearCase discusses case-sensitivity issues in depth. 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 your development project's files and directories are under ClearCase source control, you can close the **Choose Elements to Load** box by clicking **OK**. The section *Adding Elements for a New Development Task* on page 76 describes how to add elements to the VOB for a new development project.

2.5 Loading Versions of Elements into a View

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

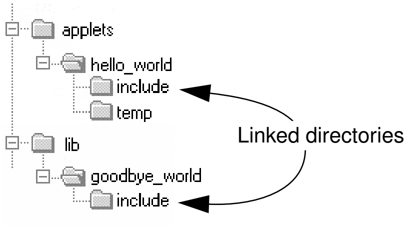
1. It uses the *version-selection rules* to select one *version* of each *element* specified by a *load rule*.
2. It copies the version into the snapshot view.
3. It records which version it copies into the view.

As ClearCase loads files and directories into your view, it shows a progress indicator in the **Snapshot View Update** dialog box. 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 8, the directory element **include** is linked.

Figure 8 VOB Link



You use the **cleartool ln** command to create VOB links. See the **ln** reference page in *ClearCase Reference Manual* for more information.

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 alternate 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 pathname, ClearCase considers other pathnames for the element as "checked out but removed." That is, to prevent you from modifying the element from multiple pathnames, ClearCase executes standard checkout behavior at only one pathname (the one from which you entered the checkout command), but does not create view-private files at other pathnames. For information about standard checkout behavior, see the **checkout** reference page in *ClearCase Reference Manual*.

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 link's pathname.

  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 pathname; 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 link target's parent directory. 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 GUIs) 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 **foo\util.c**.
2. ClearCase removes the read-only attribute from **util.c** in the **foo** directory only (which is the location from which you issued the **checkout** command).
3. You modify **foo\util.c** 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.c** in the **bar** directory.
5. You check in **bar\util.c**. Even though you checked out and modified **foo\util.c**, ClearCase does not prevent you from checking in **bar\util.c**; with a VOB hard link, **bar\util.c** is just another name for **foo\util.c**.
6. Any changes you made to **foo\util.c** 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.c** to be hijacked (even if you change attributes), because you checked out the element in the VOB.

3 Working in a View

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

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

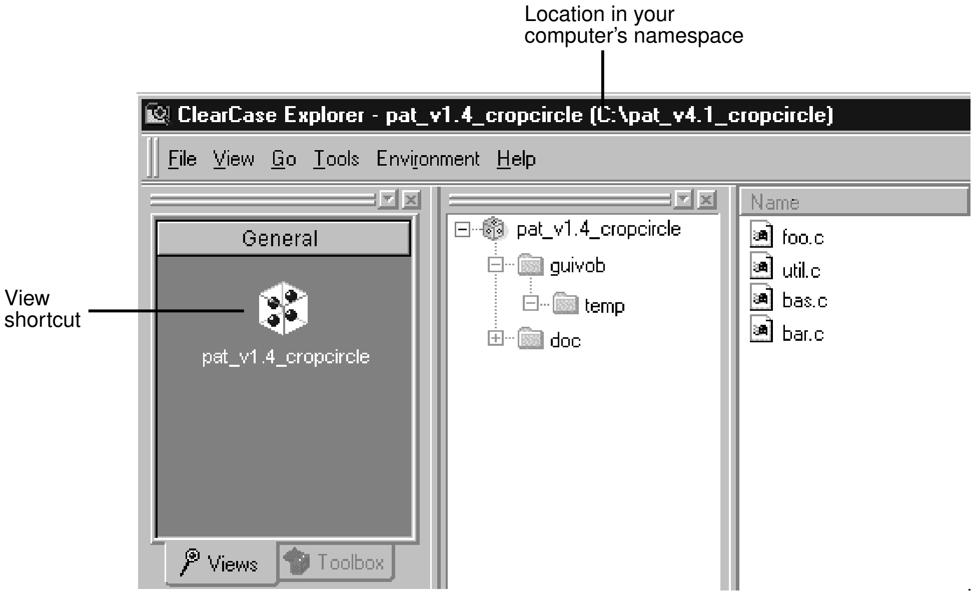
3.1 Accessing Files

The **Views** tab in ClearCase Explorer provides access to the views that you create with the View Creation Wizard. By default, ClearCase Explorer places all base ClearCase views on the General page of the **Views** tab (Figure 9).

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

To access other views from ClearCase Explorer (for example, to access a team member's view), add a view shortcut to the **Views** tab. For information on creating or modifying shortcuts in ClearCase Explorer, refer to ClearCase Explorer online help.

Figure 9 Accessing a Base ClearCase View From ClearCase Explorer



Starting Dynamic Views and Activating VOBs

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

Starting the view initiates 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 do create a ClearCase Explorer shortcut, ClearCase starts the view when you click the shortcut. Otherwise, you must start the view.

To start a dynamic view that was created from a UNIX host, you must first use the Region Synchronizer to import the view's view-tag into your Windows network region.

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's view storage directory. For more information, see *Administering ClearCase*.

To Start Dynamic Views

1. In the ClearCase Explorer Shortcut pane, click **Toolbox**. Then, click **Base ClearCase>Start View**.
2. In the **Start View** dialog box, select a view and a drive letter. Then click **OK**.

To Activate VOBs

1. In the system taskbar, click **Start**. Then click **Programs>ClearCase Administration>Mount VOB**.
2. In the **Mount VOB** dialog box, select the VOBs containing your source files.
3. Select **Reconnect at Logon** to activate the VOBs when you log on.
4. Click **OK**.

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 on assigning a snapshot view to a drive letter, see *Assigning Snapshot Views to Drive Letters* on page 87.

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. Assuming that the directory's owner has shared the directory and set up the proper permissions, use Network Neighborhood to access the view.

If you want to perform ClearCase operations in the view without adding a shortcut to the view in ClearCase Explorer, see *Registering a Snapshot View* on page 89.

Accessing Dynamic Views from Windows Explorer

You can access any view that you have started on your computer from the *dynamic-views-drive* (**M:** by default) 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 on assigning dynamic views to drive letters, see ClearCase online help.

3.2 Checking Out Files

To modify files and directories under ClearCase control, you must check them out. (Placing files and directories under source control is a separate procedure, and is described in *Adding Files and Directories to Source Control* on page 73.)

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. On the shortcut menu, click **Check Out** to open the **Check Out** dialog box.

3. In the **Check Out** dialog box, provide comments describing the changes you plan to make.

4. Determine whether you want a *reserved* or *unreserved checkout* (refer to *Reserved and Unreserved Checkouts* on page 28).

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 10, right-clicking a loaded version of a ClearCase element in the **Open** dialog box displays the ClearCase shortcut menu.

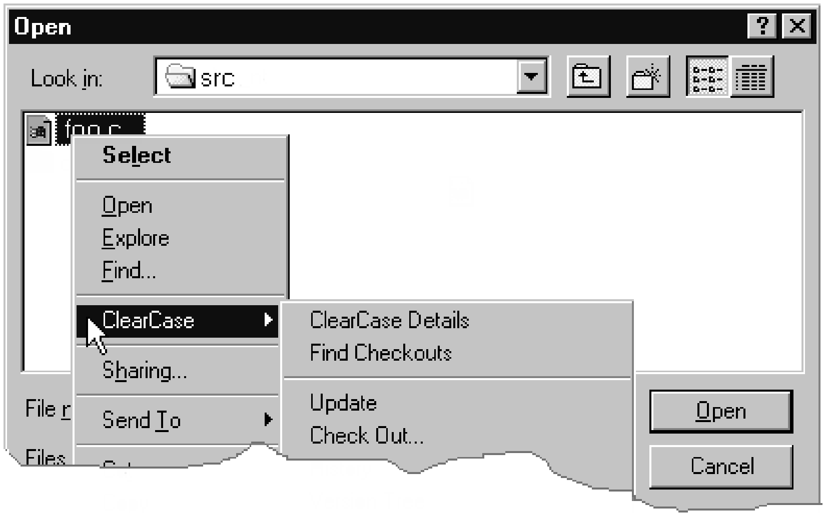
To check out a file from an **Open** dialog box:

1. In an application such as 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 10 The 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 a file's parent directory 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 on checking out files and directories from a CLI, see the **checkout** reference page in *ClearCase Reference Manual*.

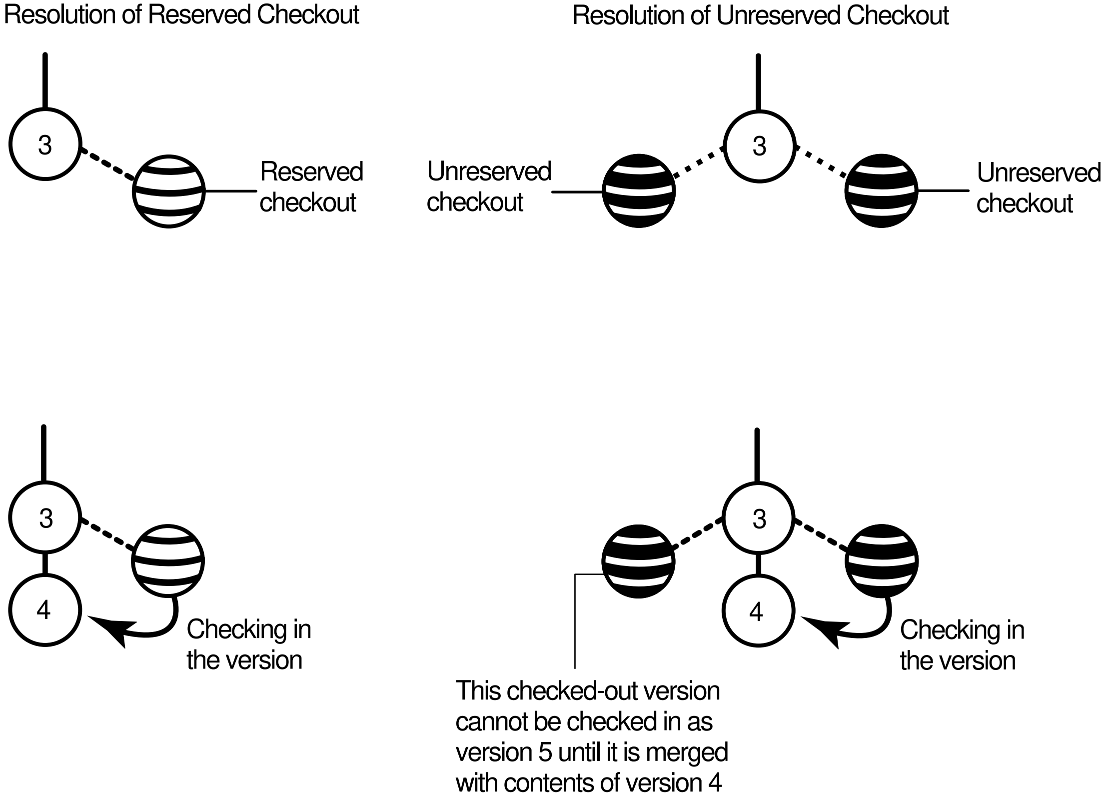
Reserved and Unreserved Checkouts

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. Your organization's development policy may determine whether to check out reserved or unreserved.

Figure 11 illustrates checked-out versions created by reserved and unreserved checkouts, and the effects of subsequent checkins.

Figure 11 Resolution of Reserved and Unreserved Checkouts



Another kind of checkout is an unreserved, nonmastered checkout, which can be used only in a replicated VOB (created with the ClearCase MultiSite product). For more information about this kind of checkout, see *Sharing Control of a Branch with Developers at Other Sites* on page 66.

To Change the Status of a Checked-Out Version

1. In the ClearCase Explorer Details pane select a checked-out version.

2. Select the **Tools** menu and choose either **Reserve** or **Unreserve**.

Setting the Default for Reserved or Unreserved Checkouts

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

1. In ClearCase Explorer, select **Tools>Options**.

2. In the **ClearCase User Options** dialog box, do one of the following in the **Check Out** box:

   - 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 checkbox, attempts to reserve a reserved checkout fail.

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

For more information unreserved, nonmastered checkouts, see *Setting the Default for Nonmastered Checkouts* on page 71.

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.

From a Snapshot View

When you use the GUI to check out a file or directory from a snapshot view, ClearCase handles the request 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.
   - Your view's config spec selects versions based on a label or a time rule, and the latest version in the VOB falls outside those parameters (Figure 12).

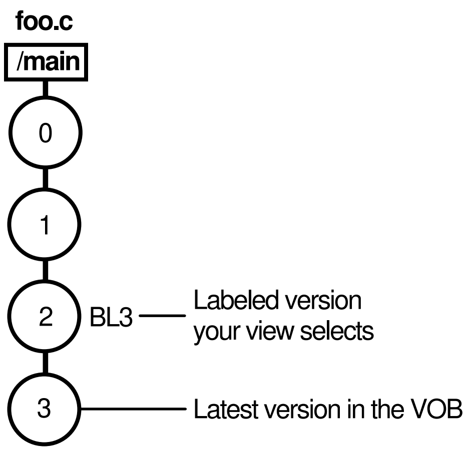
3. If you check out a version other than the one currently loaded in your view, ClearCase loads the checked-out version into your view.

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 on checking out VOB links in a snapshot view, see *Under the Hood: VOB Links* on page 18.

Figure 12 Selecting the Non-Latest Version of an Element



From a Dynamic View

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

1. If your view's version-selection rules 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 12).

   See *Merging with the Latest Version* on page 37 for information about checking in a nonlatest version.

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.

3.3 Working with Checkouts

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

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

Viewing an Element's History

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

To View an Element's History

In ClearCase Explorer, right-click an element; on the ClearCase shortcut menu, 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 colleagues?

To Compare with a Predecessor

In ClearCase Explorer, right-click a file. On the ClearCase shortcut menu, click **Compare with Previous Version**.

To Compare with a Version Other Than the Predecessor

1. 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.

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 Folder window (middle), right-click a view root 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. Click the **Include subfolders** option.
4. Click **OK**.

Prototype Builds

Typically, when you're 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 about **clearmake** or **omake**, refer to their reference pages in *ClearCase Reference Manual* and to *Building Software with ClearCase*.

3.4 Canceling Checkouts

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

1. In the ClearCase Explorer Details pane, select one or more checkouts. Then right-click.
2. On the shortcut menu, click **Undo checkout**.
3. 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 config spec's version-selection rules to select a version.

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 GUI 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 pathname of the new element) to the VOB's **lost+found** directory under names of this form:

element-name.UUID

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

```
cleartool: Warning: Object "foo.c" no longer referenced.
cleartool: Warning: Moving object to vob lost+found directory as
"foo.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 on starting the Update Tool, see ClearCase online help.

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

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 *ClearCase Reference Manual*.

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

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

For example, from a dynamic view:

> **cleartool rmelem \guivob\lost+found\foo.c.5f6815a0a2ce11cca54708006906af65**

From a snapshot view:

> **cd c:\pat_v1.4_cropcircle_sv**
> **cleartool rmelem guivob\lost+found\foo.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 element's parent directory 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.

Canceling 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 ClearCase Explorer, right-click the directory containing the deleted file.

2. On the shortcut menu, click **Find Checkouts**.

3. Complete the **Find Criteria** dialog box.

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

3.5 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 element's history. We recommend you check in a file or directory any time you want to a record of its current state.

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

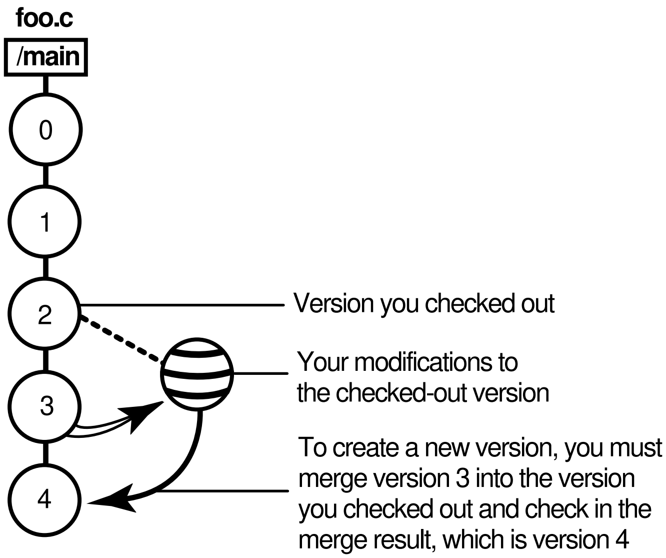
To Check In Files

1. In ClearCase Explorer, select one or more files.
2. Right-click a selected file. On the shortcut menu, click **Check In**.
3. In the **Check In** dialog box, ClearCase displays the comments you entered when you checked out the file. You can reuse these comments or modify them.

Merging with the Latest Version

If the version you checked out is not the latest version in the VOB, ClearCase requires you to merge the changes in the latest version into the version checked out in your view (Figure 13).

Figure 13 Merging with the Latest Version



The section *Under the Hood: What Happens When You Check Out a File or Directory* on page 30 describes the situations in which you may have to merge before checking in.

To Merge with the Latest Version

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 on using Diff Merge, refer to ClearCase online 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.

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 view's config spec.

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 your view's directory tree.)

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.

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 config spec's version-selection rules 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.

4 Updating a Snapshot View

The rules in your view's *config spec* 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 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.

4.1 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 View

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

1. 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 the **Advanced** tab 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 non-checked-out, nonhijacked file or directory, if the version the config spec selects is different from the version in the view, overwrite it with the version from the VOB.

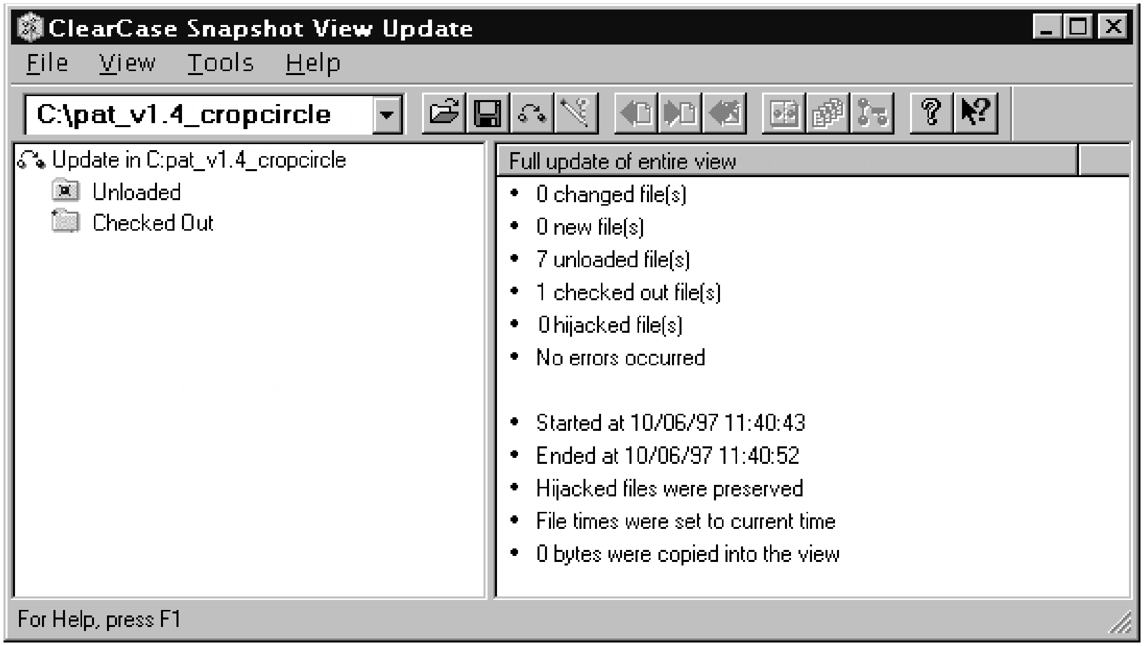
   ➢ Leave all hijacked files in the view with their current modifications.

   ➢ 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.

   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 14). For a description of this window, see the online help.

Figure 14 The Snapshot View Update Window



Updating Files and Directory Trees

To save time, you can update individual files or directories (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 34.

Tip: Finding a Set of Files

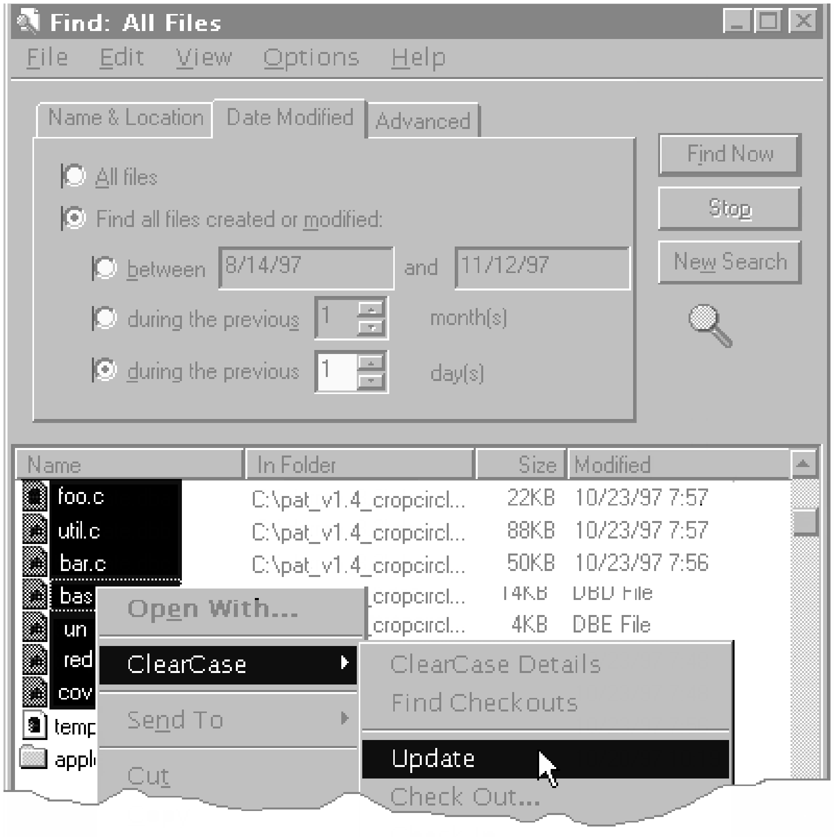
If you know that the files you want to update share a common attribute, such as a similar modification date, you can use the Find Tool in Windows Explorer to find the set of files. Then, you can update the files from the **Find** dialog box.

1. In Windows Explorer, click **Tools>Find>Files or Folders**.
2. In the **Find: All Files** dialog box, enter the path to the view or to a specific directory in the view in the **Look In** box.
3. Use the **Date Modified** and **Advanced** tabs to specify search criteria.
4. Click **Find Now**.

   The **Find: All Files** dialog box expands to display the files and directories it finds.

5. As illustrated in Figure 15, select the files you want to update. On the shortcut menu, click **ClearCase>Update**.

Figure 15 Using the Windows Find Tool to Find and Update Files



4.2 Under the Hood: What Happens When You Update a View

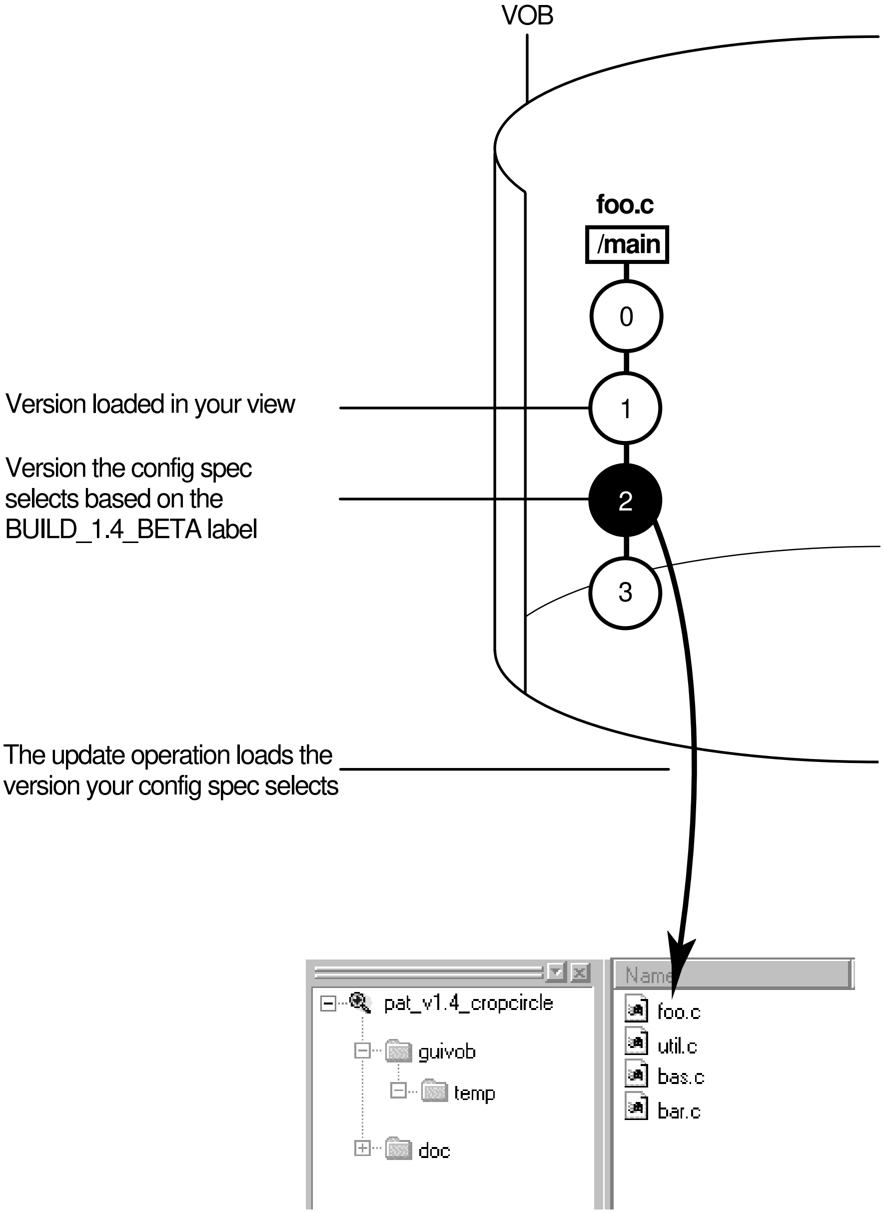
When you start an update operation, ClearCase compares the version of the elements loaded in the view with the version the config spec selects in the VOB. If the config spec selects a version in the VOB that is different from the version loaded in your view, ClearCase copies the version from the VOB into your view (Figure 16). ClearCase does not make this comparison or otherwise modify versions currently checked out to the view.

The update operation takes into account the fact that updates are not instantaneous. As ClearCase updates your view, other developers may check in new versions of elements your view's load rules 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).

Figure 16 The Update Operation



4.3 Unloading Elements

If a view's config spec 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 on removing load rules, see *To Change Which Elements Are Loaded into a Snapshot View* on page 83.

➤ 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 file's current state:

➤ 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 While Disconnected from the Network*.) You change the settings on the **Advanced** tab in the **Start Update** dialog box. Refer to *Updating the View* on page 42.

➤ 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**.

5 Working On a Team

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 GUI 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 ClearCase provides to support parallel development, a style of working in which teams use the same set of source files for different, concurrent development projects:

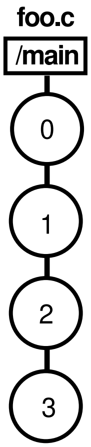
- Version trees
- Working on branches
- Merging

In addition, this chapter discusses sharing control of a branch with developers at other sites. (You do not need to read this section unless your project manager or MultiSite administrator directs you.)

5.1 The Version Tree

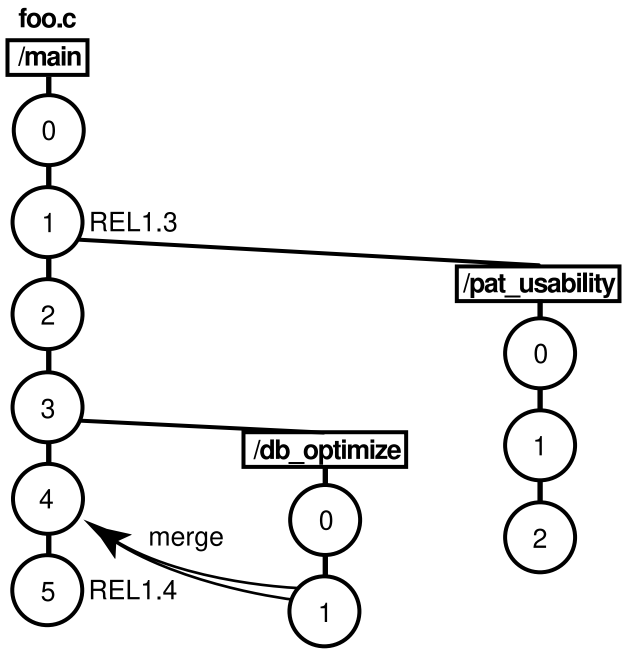
Each time you revise and check in an element, ClearCase creates a new version of the element in the VOB. Throughout this part of the book, this linear progression has been illustrated with a graphic similar to Figure 17.

Figure 17 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 until you merge. In Figure 18, **main**, **pat_usability**, and **db_optimize** are branches being used to develop different releases of the file element **foo.c** concurrently.

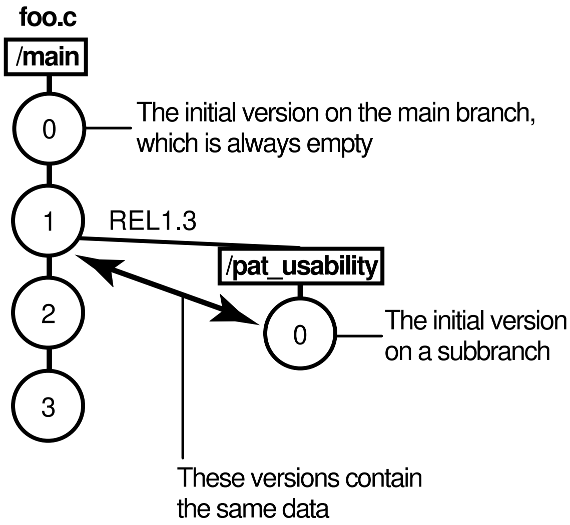
Figure 18 Version Tree of a File Element



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 19). (The initial version on the **main** branch contains no data. For more information, see *Excluding Elements* on page 83.)

Figure 19 The Initial Version on a Subbranch



5.2 Working on Branches

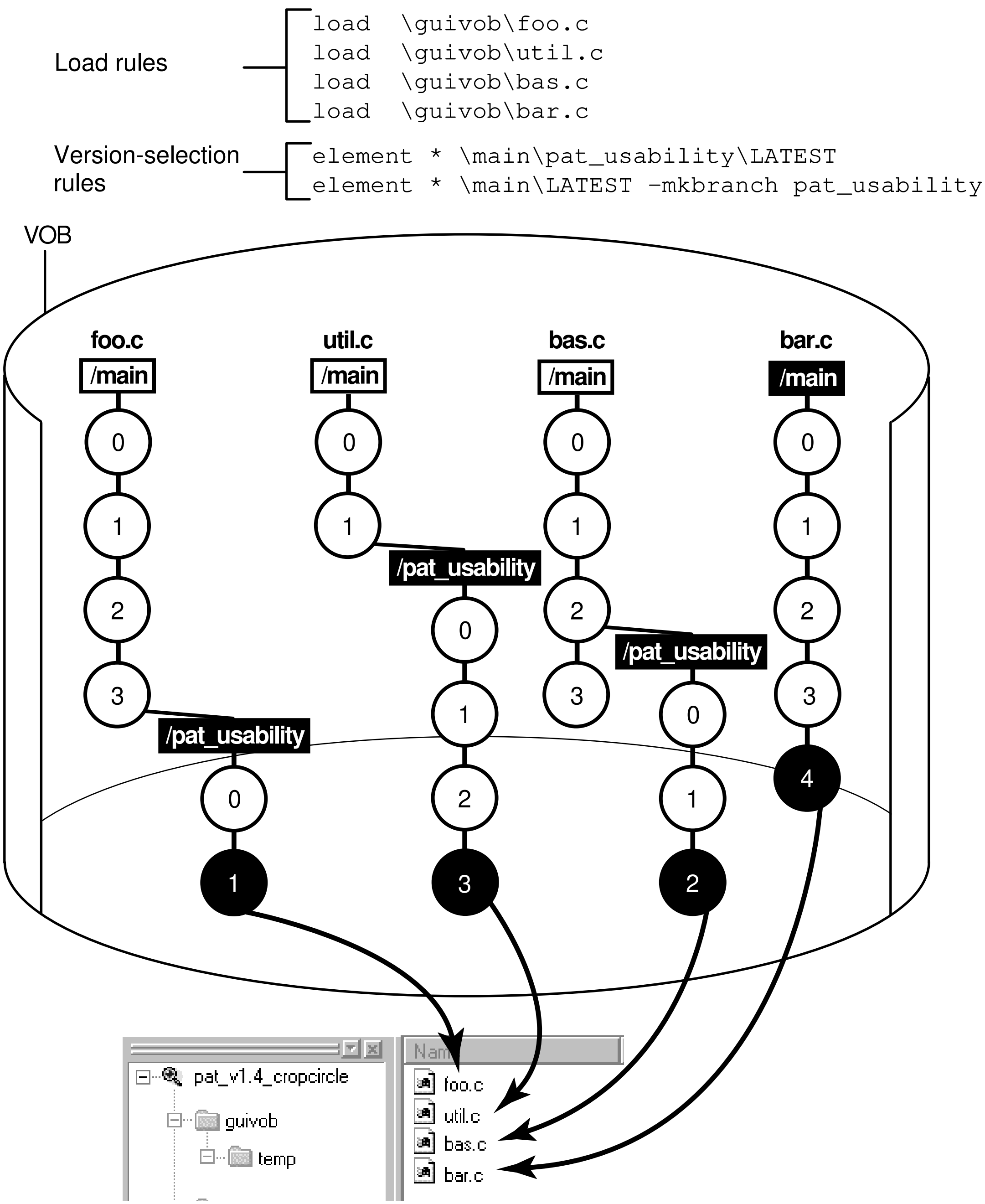
Your organization's policies may dictate that each development project use its own branch to isolate its changes from other development projects. To adhere to this policy, each member of a project team uses a view whose config spec specifies this information:

- The versions to select in the development project's specific branch. As Figure 20 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 20, because **bar.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 development project's branch (if it doesn't 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.

Figure 20 Elements Have Common Branches



For more information on branches, refer to *Managing Software Projects with ClearCase* and the **mkbranch** reference page in *ClearCase Reference Manual*.

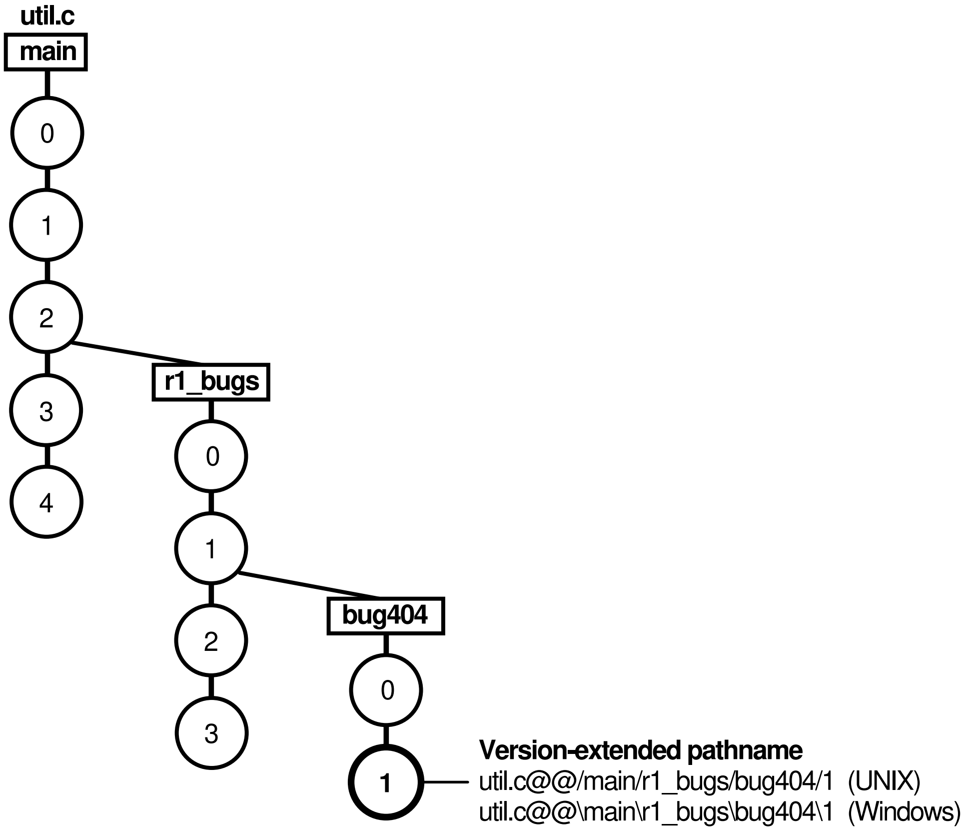
The Version-Extended Pathname

ClearCase commands and documentation use a notation to specify a version of an element on a branch. For example, **util.c@@\main\2** specifies version 2 of **util.c** on the **main** branch; **util.c@@\main\r1_bugs\bug404\1** specifies version 1 of **util.c** on the **bug404** subbranch below the **r1_bugs** subbranch, which is below the **main** branch (Figure 21).

From a command-line interface, you can use version-extended pathnames 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 pathnames.

For a full description of the syntax for version-extended pathnames, see the **ccase_pathnames** reference page in *ClearCase Reference Manual*.

Figure 21 Version-Extended Pathnames



5.3 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 versioning 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 chapter describes the following merge scenarios:

- Merging all changes made on a single subbranch
- Merging selectively from a single subbranch
- Removing the contributions of some versions on a single subbranch
- Recording merges that occur outside ClearCase

ClearCase also supports merging work from many branches to a single branch; this is typically a project manager's task and is described in *Managing Software Projects with ClearCase*.

Under the Hood: How ClearCase Merges Files and Directories

A merge combines the contents of two or more files or directories into a single new file/directory. The ClearCase merge algorithm uses the following files during a merge (Figure 22):

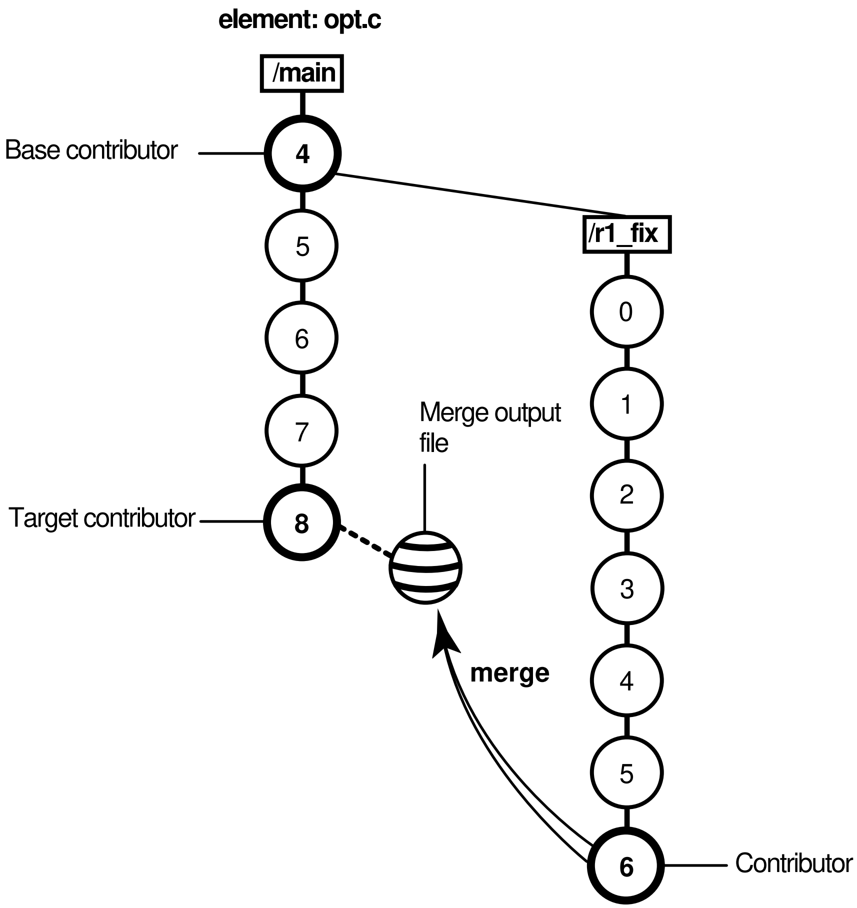
- Contributors, which are typically one version from each branch you are merging. (You can merge up to 15 contributors.) You specify which versions are contributors.

- 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 determining a base contributor, see the **merge** reference page in *ClearCase Reference Manual*.

- 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.

- 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.

Figure 22 Versions Involved in a Typical Merge



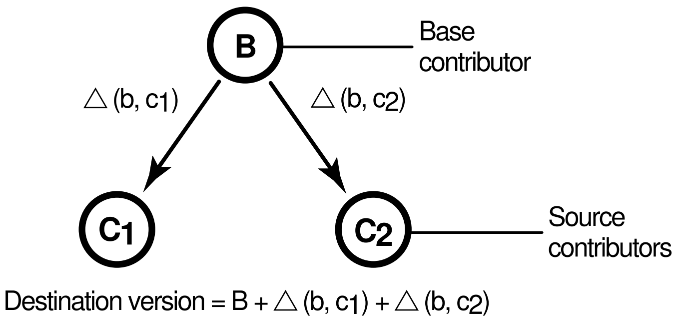
To merge files and directories, ClearCase takes the following steps:

1. It identifies the base contributor.
2. It compares each contributor against the base contributor (Figure 23).
3. For any line that is unchanged between the base contributor and any other contributor, ClearCase copies the line to the merge output file.
4. For any line that has changed between the base contributor and 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.

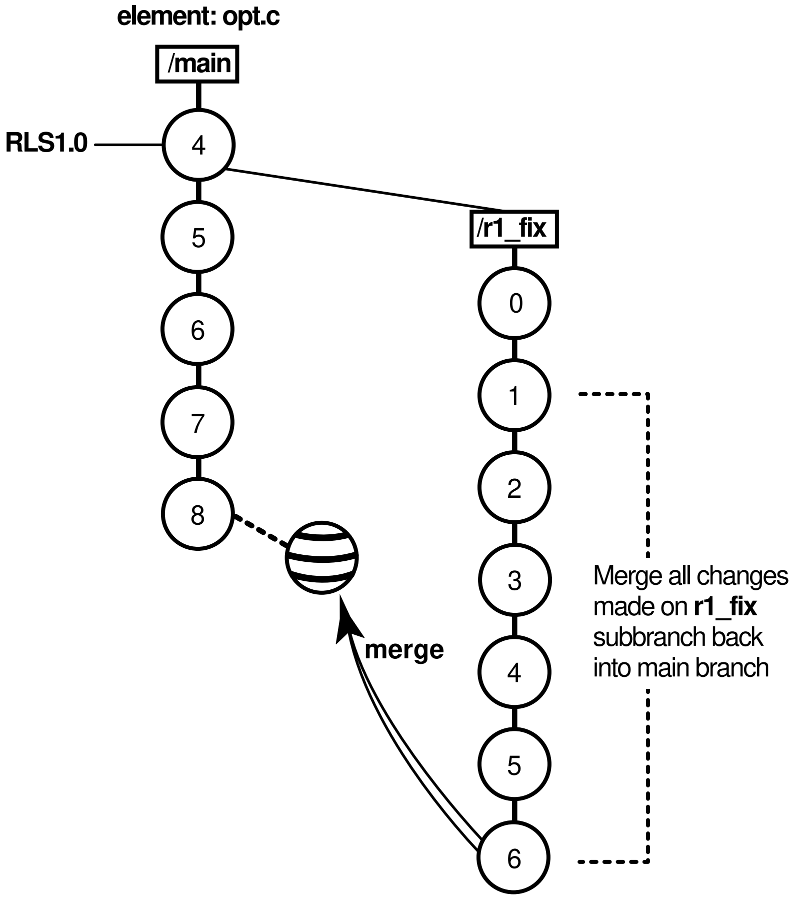
Figure 23 ClearCase Merge Algorithm



Scenario: Merging All Changes Made on a Subbranch

This is the simplest and most common case. 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 (Figure 24).

Figure 24 Merging All Changes from a Subbranch



Task Overview

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

1. Start a dynamic view or change directories to a snapshot view. The view must select the target version, which in Figure 24 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 pre-merge 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 24, 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).

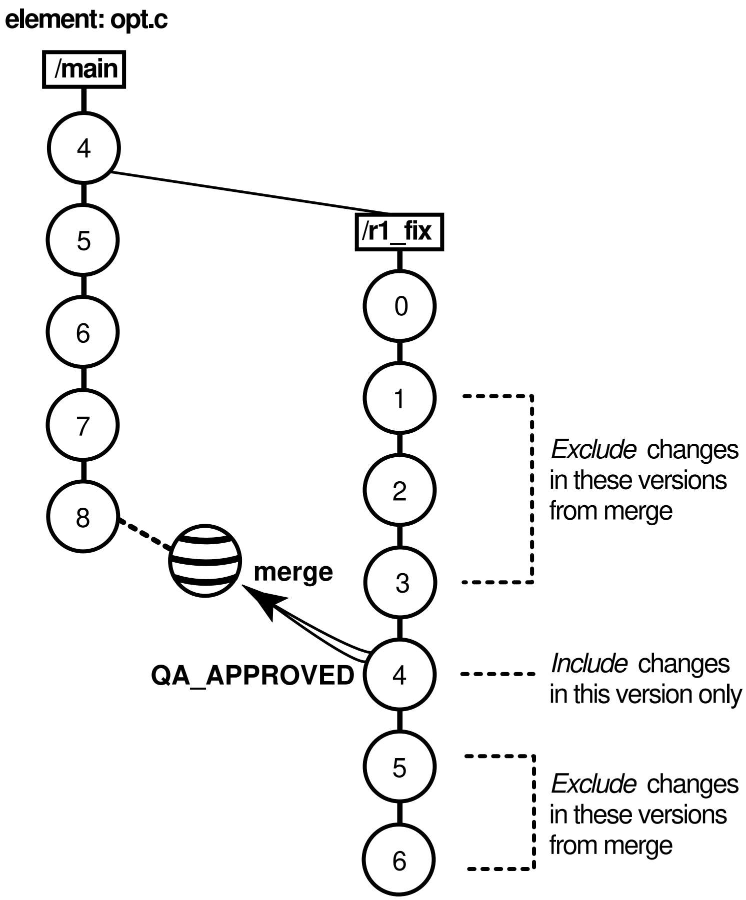
Getting More Information

For detailed information on completing this task, see ClearCase online help.

Scenario: Selective Merge from a Subbranch

In this scenario, the project manager wants to incorporate into new development several lines of code that were added in version **\main\r1_fix\4**. It's 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 (Figure 25).

Figure 25 Selective Merge from a Subbranch



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. Start a dynamic view or change directories to a snapshot view. The view must select the target version, which in Figure 25 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 the Version Tree Browser or History Browser's **Send To** command to see the contents of versions not loaded into your view. (For information about opening a version not currently in your view, see ClearCase online help.)

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:

   Z:\avob> **cleartool checkout opt.c**
   Z:\avob> **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:

   Z:\avob> **cleartool checkout opt.c**
   Z:\avob> **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 on using Diff Merge, refer to the online 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 of a version's data has been merged, not just parts of it.

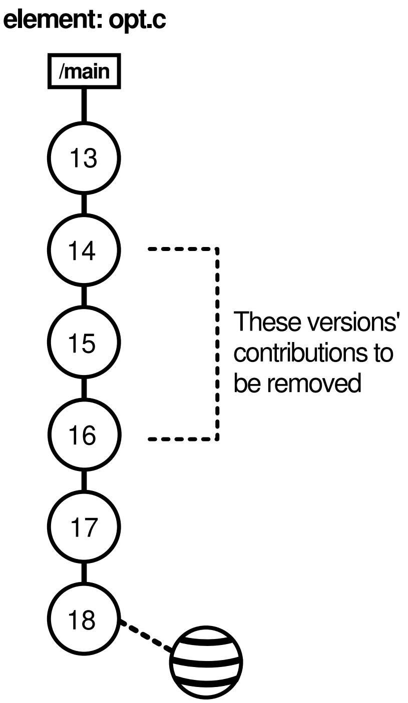
Getting More Information

For detailed information on completing this task, see the **merge** and **version_selector** reference pages in *ClearCase Reference Manual* or the ClearCase online help.

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 26).

Figure 26 Removing the Contributions of Some Versions



Task Overview

Performing a subtractive merge involves the following tasks:

1. Start a dynamic view or change directories to a snapshot view. 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 26, 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 Version Tree Browser or History Browser's **Send To** command to see the contents of versions not loaded into your view. (For information about opening a version not currently in your view, see ClearCase online help.)

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:

   Z:\avob> **cleartool checkout opt.c**
   Z:\avob> **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 on using Diff Merge, refer to online 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 on completing this task, see the **merge** and **version_selector** reference pages in *ClearCase Reference Manual* or the ClearCase online help.

Recording Merges That Occur Outside ClearCase

You can merge versions of an element manually or with any available analysis and editing tools. To update an element's version tree with a merge that occurs outside ClearCase, check out the target version, perform the merge with your own tools, and check it back in. 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've drawn 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:

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

Z:\avob> <use your own tools to merge data into checked-out version>

Z:\avob> 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 on completing this task, see the **merge** and **version_selector** reference pages in *ClearCase Reference Manual* or the ClearCase online help.

5.4 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. You do not need to 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 that replica's site 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.

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 'raleigh'. 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 'raleigh'.
Current replica is 'lexington'.
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 *ClearCase MultiSite Manual*.

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 check out the branch immediately, using a nonmastered checkout. You may have to perform a merge before you can check in your work.

The following sections describe both methods.

Waiting for Mastership to Be Transferred

For detailed information on requesting mastership from the graphical interface, see the ClearCase online help:

1. From ClearCase Explorer, click **Help>Help Topics**.
2. In the ClearCase Help window, click **Help Topics**.
3. On the **Contents** tab of the Help Contents window, select **Developing Software with Base ClearCase>Requesting mastership of a branch>To request mastership of a branch**.

You can request mastership from the Find Checkouts window, the Merge Manager, or the Version Tree Browser.

To request mastership 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
```

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: doc_lex@\doc
   ...
   ```

   In this example, your current replica is **lexington** in the VOB family **\doc**. The output of the **describe** command shows that **lexington** 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.

Checking 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 ClearCase Explorer, right-click on the element you want to check out and click **Check Out** on the shortcut menu.

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

3. If the **Confirm Version to Check Out** dialog box is open, select the **Request mastership of the branch** check box 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** on the shortcut menu.

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

c. Click the **Mastership** tab.

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

8. Use the Merge Manager to merge from 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" foo.c@@\main\integ**

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

   > **cleartool checkout –c "fix bug #28386" –unreserved –nmaster foo.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\foo.c@@\main**
   ```
   branch "\lib\foo.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're finished.

   > **cleartool checkin –nc foo.c**
   ```
   Checked in "foo.c" version "\main\65".
   ```

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

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

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

   cleartool merge –to foo.c –version \main\LATEST
   (if necessary, you are prompted to resolve conflicts)
   ```
   Moved contributor "foo.c" to "foo.c.contrib".
   Output of merge is in "foo.c".
   Recorded merge of "foo.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 clicking **Request Mastership** on the shortcut menu for nonmastered checkouts displayed by Find Checkouts or the Merge Manager, or by using the **reqmaster** command.

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. Click **Start>Programs>ClearCase>User Preferences**.

2. Click the **Operations** tab.

3. Select the **Unreserved, nonmastered if branch is mastered by another replica** check box.

When you check out an element in a replicated VOB, the **Check Out** dialog box opens with the **Unreserved, nonmastered** check box selected. 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.

6 Other Tasks

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

- Add files and directories to source control
- Move, remove, and rename elements
- Access elements not loaded into a snapshot view
- Adjust the scope of a view
- Assign snapshot views to drive letters
- Register snapshot views
- Move views
- Regenerate a snapshot view's **view.dat** file
- Access views and VOBs across platform types

6.1 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

Adding Elements to an Existing Directory Tree

NOTE: Your view's version-selection rules determine on which branch an element's first version 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. On the shortcut menu, click **Add to Source Control**.

   We recommend you select items that are the furthest 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 a File or Directory 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:

- Selecting the **Keep checked out** check box or using **mkelem** with no arguments checks out the element. Any view-private data that corresponds to the element pathname remains in your view only and is added to version 1 in the VOB when you check in (Figure 27).
- Clearing the **Keep checked out** check box or using **mkelem –ci** checks in the element, using any existing view-private data that corresponds to the element pathname as the content for version 1. Your view's config spec determines the branch on which ClearCase creates version 1.

➤ 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-pathname* exists, it is moved aside to a **.keep** file.

➤ (Replicated VOBs only) Selecting the **Make current replica the master of all newly created branches** check box or using **mkelem –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** 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 **User Options** dialog box, click **ClearCase Options**.

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

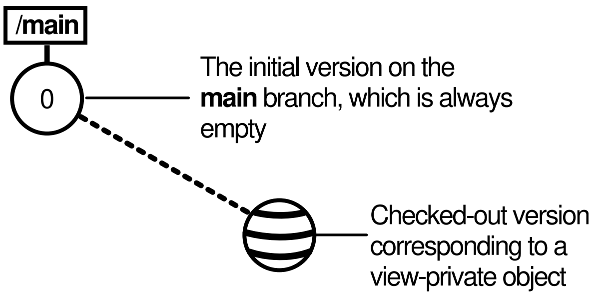
d. Click **Advanced Options**.

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

f. Click **OK**.

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

Figure 27 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.

Adding Elements for a New Development Task

NOTE: This procedure assumes that a VOB exists. For information on creating VOBs, see *Managing Software Projects with ClearCase*.

1. In ClearCase Explorer, navigate to the view you created for the development task.

2. 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.

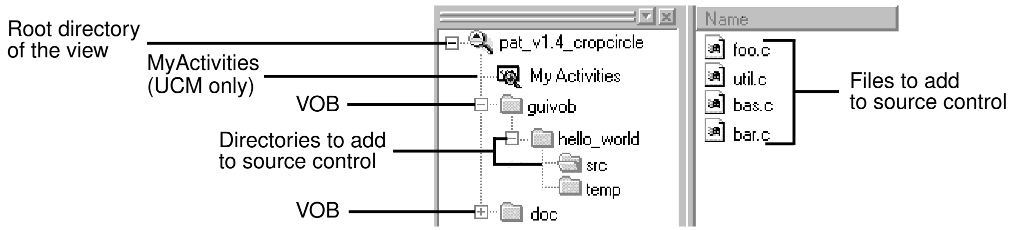
   If you work in a dynamic view, make sure that each VOB to which you want to add files and directories is accessible from your view. If a VOB isn't accessible, activate it on your computer by clicking **Start>Programs>ClearCase Administration>Mount VOB**.

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 28.

Figure 28 Directory Structure in a Snapshot View



4. In the right upper pane of 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 28, 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. On the shortcut menu, click **Add to Source Control**.

Importing Files

If you're adding a large number of files and directories to source control, use the **clearexport_ffile** command (or other **clearexport** commands) and **clearimport** command. For more information, see the **clearexport_ffile** and **clearimport** reference pages in *ClearCase Reference Manual*.

6.2 Moving, Removing, and Renaming Elements

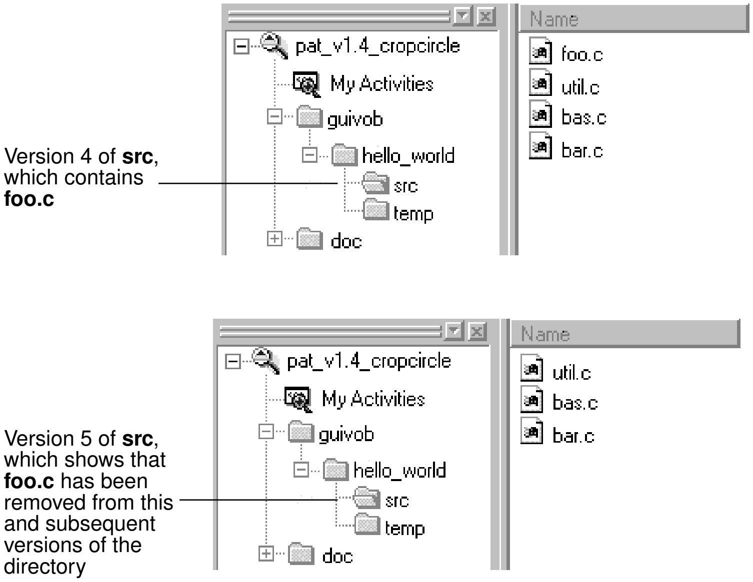
This section explains how to move, remove, and rename elements.

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 **foo.c**. If you remove **foo.c** from the **hello_world\src** directory, ClearCase creates version 5 of **\gui_vob\hello_world\src**, which does not contain the **foo.c** file element. The element **foo.c** itself is not modified (Figure 29).

Figure 29 Removing an Element Name from a Directory



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 projectwide builds.

To Move an Element Within a VOB

Do **one** of the following:

➤ In the ClearCase Explorer Details pane, drag an element to its destination directory in the Folder Window.

➤ In the ClearCase Explorer Details pane, right-click an element and click **Cut**. In the Details pane, right-click the destination directory and click **Paste**.

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, you are advised to 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 ClearCase Explorer Details pane, right-click an element and click **Delete**.

Other Methods for Removing Elements

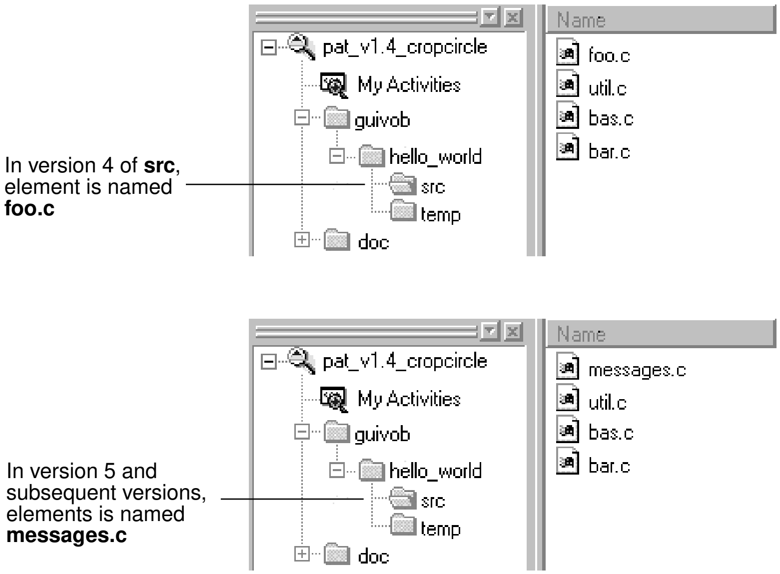
Removing an element from its parent directory does not affect the element itself, but two other types of a removal operation do irrevocably affect an element, and we recommend that you be very conservative in using these operations:

- Removing a version from an element's version tree. For more information, see the **rmver** reference page in *ClearCase Reference Manual*.
- Removing an element from a VOB. For more information, see the **rmelem** reference page in *ClearCase Reference Manual*.

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 30).

Figure 30 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 ClearCase Explorer Details pane, right-click an element and click **Rename**.
2. In the Details pane, type a new name for the element
3. Press ENTER.

6.3 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 might want to check periodically whether the functional specifications have been modified by reviewing the element's ClearCase history.

Listing All Elements in the VOB Namespace

You can set up ClearCase Explorer to provide a directory listing of nonloaded file and directory elements. 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's history, version tree, and properties sheet, and compare versions of the element.

To See Nonloaded Elements from ClearCase Explorer

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

2. In the **Options** dialog box, under **Snapshot View Options**, select **Show Unloaded Snapshot View Elements**.

3. Click **OK**.

   The Details pane uses an icon and the State column to indicate which versions are not loaded.

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 version's shortcut menu 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, refer to Windows online 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 ClearCase Explorer Details pane, in the Version column, note the *version-extended pathname* 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 using the following syntax:

   get –to *filename version-extended-pathname*

   For example, **get –to foo.c.previous.version foo.c@@\main\v3.1_fix\10** copies **foo.c@@\main\v3.1_fix\10** into your view under the name of **foo.c.previous.version**.

6.4 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, click **Toolbox.** Then click **Base ClearCase>Edit View Properties**.
2. In the **Edit View Properties** dialog box, select the view and click **Edit**.
3. In the **Properties** dialog box, click the **Load Rules** tab.
4. On the **Load Rules** tab, click **Edit Load Rules**.
5. In the **Choose Elements to Load** dialog box, click **Add** to load another element into the view, or click **Remove** to unload a file or directory from the view.
6. Click **OK**.

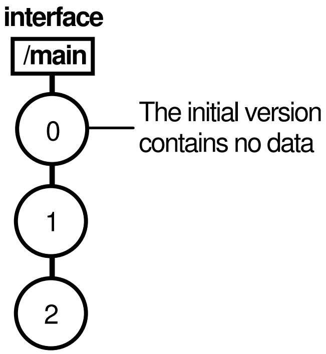
ClearCase starts the update operation to match the modified config spec. For more information, see *Under the Hood: What Happens When You Update a View* on page 46.

Excluding Elements

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

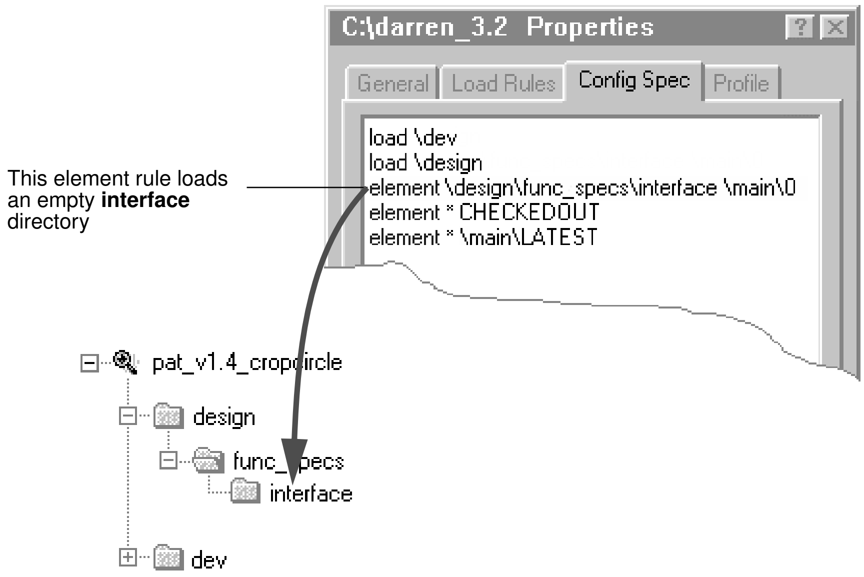
To exclude some elements, you can use an element rule in the *config spec* that selects an element's initial version on the **main** *branch*. For all ClearCase elements, the initial version on the **main** branch (illustrated in Figure 31) contains no data. (See Chapter 5, *Working On a Team* for a description of branches.)

Figure 31   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 32). 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.

Figure 32   Excluding the interface Directory



To Load an Empty Version of an Element

1. In the ClearCase Explorer Shortcut pane, click **Toolbox**. Then click **Base ClearCase>Edit View Properties**.
2. In the **Edit View Properties** dialog box, select the view and click **Edit**.
3. In the **Properties** dialog box, click the **Config Spec** tab.
4. On the **Config Spec** tab, click **Edit** to make the rules on the **Config Spec** tab editable.
5. As illustrated in Figure 32, 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.
6. In the Properties dialog box, click **OK**.

Activating or Deactivating VOBs

Activating a VOB on your computer makes its files and directories available to your dynamic views. Deactivating a VOB frees your computer's resources.

To Activate VOBs

1. In the system taskbar, click **Start**. Then click **Programs>ClearCase Administration>Mount VOB**.
2. In the **Mount VOB** dialog box, select the VOBs containing your source files.
3. Select **Reconnect at Logon** to activate the VOBs when you log on.
4. Click **OK**.

To Deactivate VOBs

1. In the system taskbar, click **Start**. Then click **Programs>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, refer to *Working on Branches* on page 53. Depending on your organization's development policies, your view's version-selection rules may select versions on a specific branch.

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

2. In the **Edit View Properties** dialog box, select the view and click **Edit**.

3. In the **Properties** dialog box, click the **Config Spec** tab.

4. On the **Config Spec** tab, 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.

To learn more about *version-selection rules*, refer to the **config_spec** reference page in *ClearCase Reference Manual* or to *Managing Software Projects with ClearCase*.

6.5 Assigning Snapshot Views to Drive Letters

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

Depending on your computer's configuration, two methods are available:

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

Assign Only the Root Directory. You must assign only the snapshot view's root directory to a drive letter. 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 view's root to a drive letter, ClearCase cannot find the view's **view.dat** file and assumes that the current directory is not a snapshot view.

Creating a Shared Directory and Assigning It to a Drive Letter

From Windows NT hosts, you can create a shared directory and assign it to a drive letter. With this method, you can access a view through Network Neighborhood, but the performance is slightly slower than if you use the **subst** command to assign the view to a drive letter.

From Windows 95 or Windows 98 hosts, you can assign a shared directory to a drive letter 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 letter. To assign your own Windows 95 or Windows 98 view to a drive letter, you must use the **subst** command.

To Create a Shared 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 letter on their computers), you must enable file sharing before using this procedure.

1. In Windows Explorer, click the snapshot view's 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's root directory.)

4. Click **OK**.

To Assign the Root Directory to a Drive Letter

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

1. In Network Neighborhood, navigate to the view's root directory.

2. Right-click the folder; on the shortcut menu, select **Map Network Drive**.

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

4. Click **OK**.

Using the subst Command

Assigning a view to a drive letter with the **subst** command provides slightly better performance than making the snapshot view a shared directory; however, only shared directories are accessible through Network Neighborhood.

1. Open a command shell.

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

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

   maps the **pat_v1.4_cropcircle** directory to the **Y:** drive. Assign only the view's root directory to a drive letter.

For more information on the **subst** command, type **help subst** in a command shell.

6.6 Registering a Snapshot View

When you create a snapshot view or click on a view shortcut in ClearCase Explorer, ClearCase registers it in your Windows User Profile. The integration with Windows Explorer recognizes files and directories below a registered snapshot view as ClearCase objects.

If a view is not registered in your Windows User Profile and you access the view through Network Neighborhood, the Windows Explorer integration does not recognize the files or directories as ClearCase objects. If you want to perform ClearCase operations in this view, you must either create a view shortcut in ClearCase Explorer or, if you want to access the view only through Windows Explorer, register the view.

NOTE: You cannot register a snapshot view that was created from a UNIX host.

For example, if you use Windows Explorer to navigate to a team member's view on a different computer and right-click a file in the view, ClearCase options are not available. To perform ClearCase operations on the files in a team member's view, register the view 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.

   When you right-click the root directory of the view, ClearCase registers the view by adding an entry to your Windows User Profile.

6.7 Moving Views

This section discusses the following tasks:

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

For information on changing a view-tag, see the **mktag** reference page in *ClearCase Reference Manual*.

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 snapshot view's directory tree of loaded elements and view-private files. You can move the view to a different computer, but the computer must run a Windows operating system.

CAUTION: If the view storage directory is located below the root directory of the view, **do not use** standard Windows commands to move the snapshot view. Instead, see *Moving a View Storage Directory*.

To Find the Location of the View Storage Directory

1. In ClearCase Explorer, right-click the view's root directory.

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

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

Update After Moving

After moving a snapshot view, you must update any view shortcuts in ClearCase Explorer that use the old pathname 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. 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's **view_server** process 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. *Administering ClearCase* describes the procedure for moving view storage directories.

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 *Administering ClearCase*.

6.8 Regenerating a Snapshot View's 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-pathname*

   For example:

   > **ccperl c:\\atria\\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.

6.9 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 Platform Types

Your 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 *ClearCase Reference Manual*.

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 view's behavior.

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). Even though 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: it does not create symbolic links if the load rules encounter a VOB symbolic link, and you can issue ClearCase commands for the view only from Windows hosts (ClearCase hosts on UNIX will not recognize the directory tree as a snapshot view).

Accessing Views Across Platform Types

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 Platform Types*.

2. Import the UNIX view's view-tag 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 Platform Types

Your administrator sets up VOBs on Windows or UNIX hosts and creates *VOB-tags* in each ClearCase network region that needs to access the VOBs. (See *Administering ClearCase* for information on registering UNIX VOB-tags in a Windows network region.) Then, from any ClearCase host on Windows or UNIX 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 1 summarizes your options for accessing VOBs across platform types.

Table 1 Accessing ClearCase VOBs Across Platform Types

Platform Type of Your ClearCase Host	Platform Type 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 Platform Types

If developers check in source files from views created on both Windows and UNIX hosts, consider creating your views in insert_cr or strip_cr text mode. The text modes change how a view manages line terminator sequences. For more information about view text modes, refer to *Administering ClearCase* or ClearCase online help.

A

Working in a Snapshot View While Disconnected from the Network

If you need to work with your source files from a computer that is disconnected from the network of 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
- ➤ Re-connecting 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 ClearCase Web interface. Talk to your ClearCase administrator to see whether the ClearCase Web interface 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 online help.

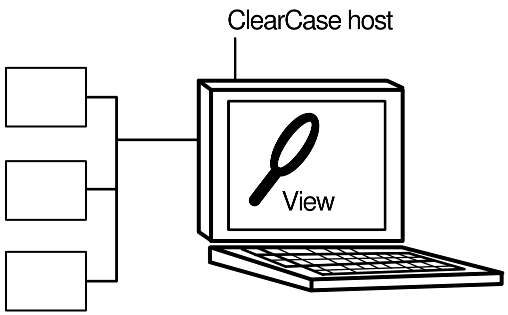
A.1 Setting Up a View for Your Hardware Configuration

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 33).

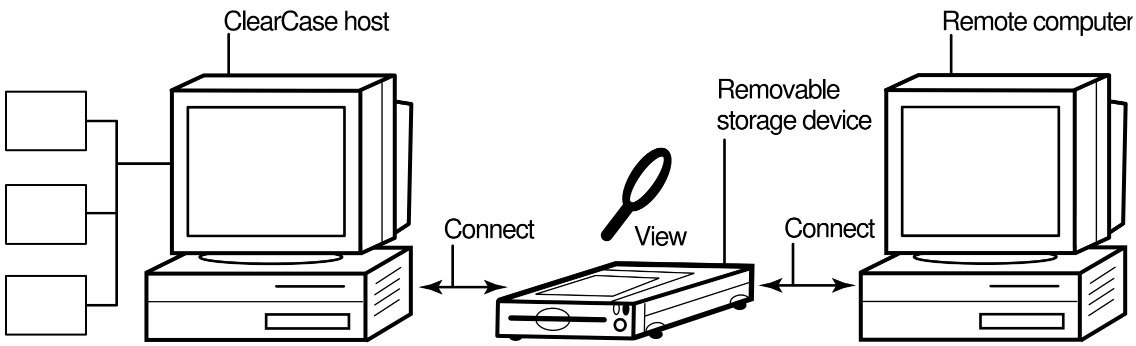
Figure 33 View on a Laptop



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 34).

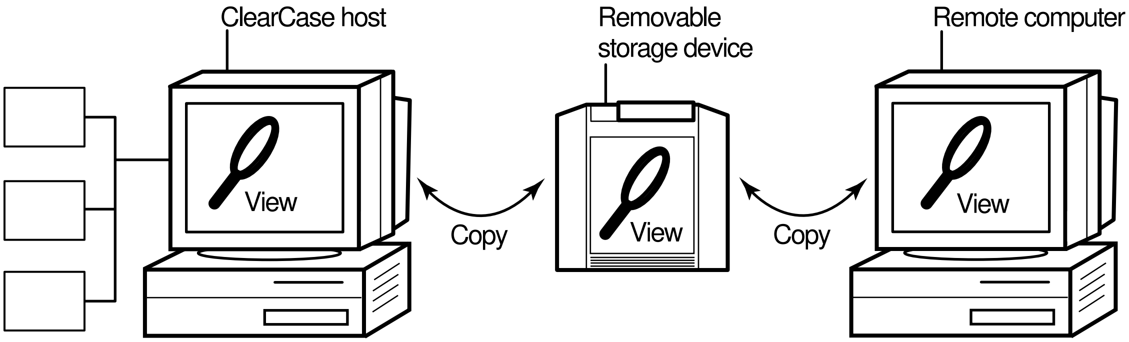
Figure 34 View On a Removable Storage Device



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 35).

Figure 35 Copy the View



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

Under the Hood: Location of the View Storage Directory 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 a view's **view_server** process runs on the host that 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.

A.2 Preparing the View

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

- ➤ Update the view to establish a checkpoint. (For information on updating the view, see *Updating the View* on page 105.)
- ➤ Check out the files you expect to modify. After you're disconnected from the network, you cannot check out files, although there are workarounds. (See *Hijacking a File* on page 100.)

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.

A.3 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
- ➤ Copying the view

Hardware Configuration: View on a Laptop

If the view is located on a laptop, you must deactivate the ClearCase integrations with Windows Explorer 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 from Windows Explorer) your host must connect to the ClearCase license server. 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 ClearCase license server locks for several seconds until it returns an error message. Although the application doesn't fail, such interruptions are unnecessary.

To Deactivate ClearCase Integrations

1. In Control Panel, click **ClearCase**.

2. Click the **Options** tab and clear the **Enable ClearCase network operations** check box.

3. Disconnect the laptop from the network.

Deactivating the integrations removes ClearCase commands from the 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's time stamp option on the **Advanced** tab of the **Start Update** dialog box. For more information, refer to Step #3 of *Updating the View* on page 105.

  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.

A.4 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: How ClearCase Determines Whether a File is Hijacked* on page 105.

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're ready, check in your changes.
- Undo the hijack. For more information, see *Undoing a Hijack* on page 104.

To Hijack a File

1. In Windows Explorer, right-click the file you want to hijack.
2. On the shortcut menu, click **Properties** to display the file's property sheet.
3. On the **General** tab, clear the **Read-Only** check box.

Finding Modified Files While Disconnected

You can use Windows Explorer to find all files that have been modified after a specified date.

1. In Windows Explorer, click **Tools>Find>Files or Folders**.
2. In the **Find: All Files** dialog box, type the path to the view or to a specific directory in the view in the **Look In** box.
3. Use the **Date Modified** tab to define your search criteria.
4. Click **Find Now**.

   The **Find: All Files** dialog box expands to display the files and directories it finds.

A.5 Connecting 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're using 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. In Control Panel, click **ClearCase**.
2. Click the **Options** tab and select the **Enable ClearCase network operations** check box.

Hardware Configuration: View on a Removable Storage Device

If you're using 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.

A.6 Using the Update Tool

When you're 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
- ➤ Undoing a hijack
- ➤ Choosing other ways to handle hijacked files

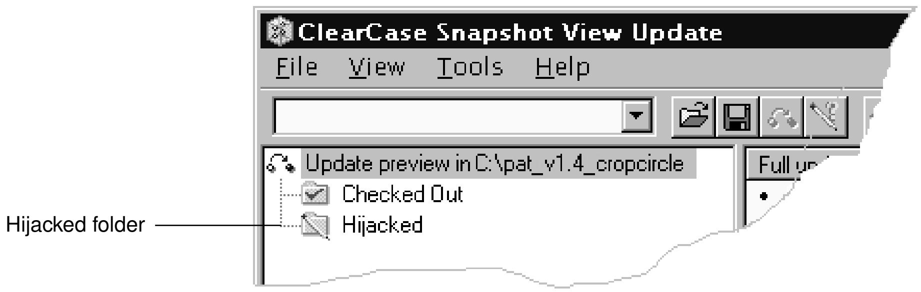
To Find Hijacked Files

1. In ClearCase Explorer, right-click the root directory of the snapshot view.

2. On the shortcut menu, click **Find Modified Files**.

3. If any hijacked files are in your view, the ClearCase Snapshot View Update window displays a folder in the left pane titled **Hijacked** (Figure 36).

Figure 36 Hijacked Files in the Results Window



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 ClearCase Snapshot View Update window, right-click a hijacked file.
2. On the shortcut menu, click **Compare with Original Version**. For information on using the Diff Merge tool, see the online help.

Checking Out a Hijacked File

To keep the modifications in a hijacked file, check out the file:

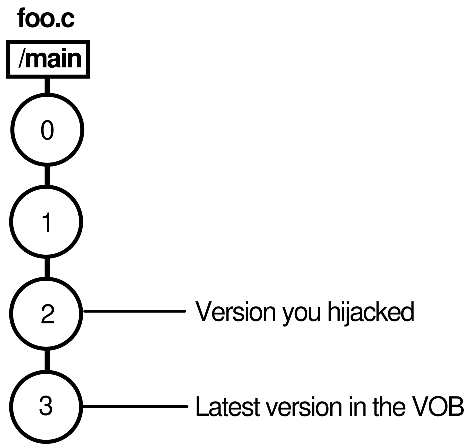
1. In the right pane of the ClearCase Snapshot View Update window, right-click a hijacked file.
2. On the shortcut menu, click **Check Out**.
3. ClearCase treats a checked-out hijacked file as it does any other checkout.

When you're ready, you can check in the file or, if necessary, merge your changes with a version in the VOB.

You May Be Prompted to Merge

If you're 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 37), you have to merge the hijacked file with the newer version in the VOB at checkout.

Figure 37 Hijacked Version May Not Be the Latest Version



To Merge with the Latest Version

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 on using Diff Merge, refer to ClearCase online help. After the merge, ClearCase prompts you to check in the file.

Undoing 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 ClearCase Snapshot View Update window, select one or more hijacked files.

2. Right-click the selected files, and on the shortcut menu, click **Undo Hijack**.

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, refer to Step #3 in *Updating the View* on page 105.

Under the Hood: How ClearCase Determines Whether a File is Hijacked

To keep track of file modifications in a snapshot view, ClearCase stores a loaded file's size and last-modified time stamp (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 a non-checked-out file's read-only attribute alone 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

See *Updating the View* for more information.

Updating the View

1. In ClearCase Explorer, select the snapshot view's 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 three choices:

   - Leave hijacked files in place
   - Rename the hijacked files and load the version from the VOB
   - Overwrite hijacked files with the version the config spec selects in the VOB

4. To start the update, click **OK**.

Index

E

F

G

H

I

L

V

W