User's Guide

Versions® 2.0

The Intuitive Version Control Solution™

StarBase®

Versions 2.0a

Limited Warranty

StarBase® Corporation guarantees your satisfaction with this product for a period of 30 days from the date of original purchase. If you are unsatisfied with Versions within that time period, contact StarBase Corporation for a full refund less shipping and handling. StarBase Corporation warrants that the program will perform in substantial compliance with the online and written documentation supplied with the software product. If a significant defect in the product is found, the Purchaser may obtain a refund. In no event will such a refund exceed the purchase price of the product, exclusive of freight charges.

EXCEPT AS PROVIDED ABOVE, STARBASE CORPORATION DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE PRODUCT. SHOULD THE PROGRAM PROVE DEFECTIVE, THE PURCHASER ASSUMES THE RISK OF PAYING THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION AND ANY DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES. IN NO EVENT WILL STARBASE CORPORATION BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION AND THE LIKE) ARISING OUT OF THE USE OR THE INABILITY TO USE THIS PRODUCT EVEN IF STARBASE CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
Use of this product for any period of time constitutes your acceptance of this agreement and subjects you to its contents.

Acknowledgments

The following people converted time, energy, and creativity into this product:

Nancy Allis
Charles Chien
Alan Ciemian
Bryan Flynn
Eric Furlong
Danielle Gutfinger
Joan Isenbarger
Alan Kucheck
Won Lee
David Mackinder
Betsy Malone
Basil Maloney
Jon Marinello
Tim May
Stephen Reynolds
Wojciech Siedlecki
John Sileski
Brian Stiles
Naoma Stow
Tu Vo
Tom Woodward

and

Mike Benson

Copyright Notice

Table of Contents

Preface

Use the installation card to install Versions on your workstation. This manual contains the information you need to use Versions. Versions can be integrated with StarTeam products. Evaluation copies of the StarTeam products are also included on the CD-ROM.

What's on the CD-ROM

The Versions CD-ROM contains the following:

\Versions	The files necessary to install Versions onto your system.
\StarTeam Pro	The files necessary to install an evaluation copy of StarTeam Workstation Professional onto your system.
\StarTeam	The files necessary to install an evaluation copy of StarTeam Workstation onto your system.
\Server	The files necessary to install an evaluation copy of StarTeam Server, which provides client/server WAN access to StarTeam projects.
\WConnect	The files necessary to install an evaluation copy of StarTeam Web Connect, which makes it possible to connect to StarTeam projects via an Internet web browser.

\Tutor	The Versions and StarTeam Walkie-Talkies, a series of .avi files that introduce you to the respective Workstation interfaces and functionality. These .avi files are not automatically installed onto your system and require a sound card.
\Acrobat	The files necessary to install the Acrobat Reader on your system. The Acrobat reader allows you to read the .pdf files in \Docs.
\Docs	Online versions of this manual and the manuals for StarTeam Workstation, StarTeam Server, StarTeam Web Connect, and Software Configuration Management Overview which can be read using the Acrobat Reader. The online version of this manual is named Versions.pdf.

As part of the installation for Versions, you can install the **StarDraw Sample Application** that contains the StarDraw sample application files.

Most of the directories mentioned here have a readme.wri file. The readme.wri in the root directory gives an overview of the StarBase products.

Accessing Electronic Manuals

You have access to the *Versions User's Guide* in electronic form. This is in addition to the context sensitive help automatically installed with Versions and available from the Help menu.

If you do not want to install the online documentation, you can still access it on the Versions CD-ROM. In the \Docs directory are:

versions.pdf	*Versions User's Guide*
starteam.pdf	*StarTeam Workstation User's Guide*
server.pdf	*StarTeam Server User's Guide*

wconnect.pdf	*StarTeam Web Connect User's Guide*
scm.pdf	*StarTeam Configuration Management Overview*

Because the online manual for Versions and StarTeam Workstation is in Adobe Portable Document Format (.pdf), you must have the Adobe Acrobat Reader installed. If you do not already have Adobe Acrobat Reader installed, follow these instructions:

1. With the Versions CD loaded, run:

   `x:\acrobat\ar32e30.exe`

   where x: is the drive letter of your CD-ROM drive.

   This program allows you to read .pdf files.

   (This directory also contains a ReadMe file that gives more details regarding the Adobe Acrobat Reader.)

2. Follow the on-screen instructions to finish the Acrobat Reader installation.

When installed, the Acrobat Reader is associated with .pdf files, allowing you to access the electronic copies of the Versions and StarTeam users guides.

Video Tutorials

The Versions CD-ROM includes a video tutorial to teach your team about Versions. However, listening to the tutorials requires a multi-media sound card. To run these tutorials, click **View Tutorials** in the master setup program (\setup.exe on your Versions CD-ROM). Then click the name of a tutorial.

StarDraw Sample Project

The Versions CD-ROM also includes a sample Versions project, called StarDraw, that is complete and ready to use. It contains a Visual C++ sample application and related materials.

The StarDraw sample project is installed on your hard drive as part of the Typical installation. To use this project, double-click on the **StarDraw Sample Project** icon. Log in using the name **NAllis**, and take a test drive.

Online Help

For a quick explanation about setting up a project, adding files, checking files in and out, etc., please see the online help file versions.hlp. Select the Contents tab, double-click **Versions** then **Quick Start**.

A Special Note for PVCS Users

The PVCS Import utility installed with Versions works and was tested with projects created using the 32-bit version of PVCS 5.2.1. PVCS Import relies on your workstation having a valid copy of PVCS Version Manager installed. If you encounter errors while running PVCS Import, make sure you can successfully run PVCS Version Manager and that the PVCS Version Manager directory is in your workstation's PATH. See Chapter 14, *"Importing Projects from PVCS"* for more information.

Conventions

Convention	Meaning
Select **File** ⇨ **Run...**	Menu selections will look like this. The arrow (⇨) indicates a drop-down menu selection. In this case, select File from the menu bar, then select Run... from the drop-down menu.
☆ **NOTE**	Note indicates important or supplemental information.
☆ **TIP**	Tip indicates suggestions that may be helpful.
`Fixed-Space Type`	Information that you need to type and messages from the system are displayed in fixed-space type.
	Icon that identifies an administrative function.

What's New in Release 2.0a?

Versions offers several new features:

Subproject branching	Enhanced subproject branching and merging enables recursive branching using either the latest version, a version label or date designation. See Chapter 3, "Managing Subprojects."
MRG Command	The merge command's new /is command-line option allows you to include subdirectory merges with particular subprojects and all of its children. See Chapter 3, "Managing Subprojects" and Chapter 13, "Using the Command-line Interface."

Mandatory check-in	This administrative feature forces a check-in reason for file changes and keeps team member up to date on why a file changed. Maximum check-in reason length was also raised from 255 characters to 20K to maintain a more detailed file change history. See Chapter 2, “Managing Projects.”
Log keyword	Lines expanded using the Log keyword can be prefixed with comment characters appropriate to file type. See Chapter 4, “Managing Files.”

Chapter 1
Introducing Versions

Versions facilitates team collaboration through the use of tightly integrated components commonly used during product development: version control, defect tracking, and threaded conversations. In addition, it provides an audit log of the actions performed on project components and information about the team members performing those actions. This combination:

- Allows you to store everything related to a project in one place using just one application.
- Allows you to access information and collaborate via LAN.
- Improves teamwork on a project.
- Saves maintenance time and makes testing easier because of the tight integration between version control and defect tracking.
- Keeps track of who did what and when. For example, when the audit log shows that someone checked in a file that was later determined to have a regression, you can undo the problem by checking out the previous version.
- Is targeted for Windows 95 and Windows NT

Using Version Control

If you think that you have never used version control, you are probably wrong.

Maybe you used different filenames for different versions of the same document. For example, you might have named a file rpt0908.doc on September 8th and changed it to rpt0910.doc on September 10th—just so you could keep both versions. This is version control. It allows you to return to the September 8th version if the changes you made September 10th become obsolete.

Many popular software packages maintain a single extra copy of your work in a .bak file. This "backup" file contains the previously saved copy of your work so you always have access to the two latest versions of the file. This scheme has proven valuable to most people at one time or another. It allows you to recover the previous version of a file if you accidentally delete or damage the original. This, too, is version control.

If you are new to version control, this introduction gives you the basics. If you are not new to version control, it explains what features Versions offers and how they are implemented.

Versions allows you to maintain any number of previous versions in a special directory called the vault. As far as you are concerned, the file has only one name. However, Versions gives each copy a unique name and keeps track of these unique names for you. The vault is a directory on your hard drive or, more likely, a network drive. Other than being sure never to delete the directory or any file in it, the vault is of no concern to you.

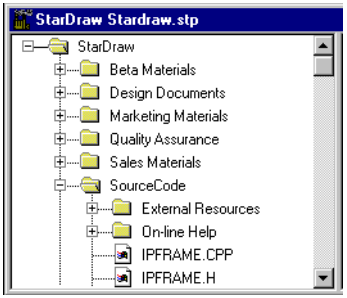
Organizing the Project Hierarchy

Versions provides more than just version control of individual files. It allows you to organize entire groups of related files—files for source code, documentation, marketing materials, pricing spreadsheets, and so forth.

Versions provides a project window that displays the files in a hierarchy. Usually the project administrator creates subprojects for a project. Then you and other users create files to add to those subprojects. These form a tree consisting of the root subproject, its

subprojects, and the files in those subprojects. You select a file from the project tree when you want to view the file's history, which lists the versions of that file currently stored in the vault.

The following figure represents the StarDraw project. The open folder at the top is the root subproject. The folders within it, such as those named Beta Materials and Design Documents, are subprojects. The page icons (in front of IPFRAME.CPP and IPFRAME.H) represent individual files.



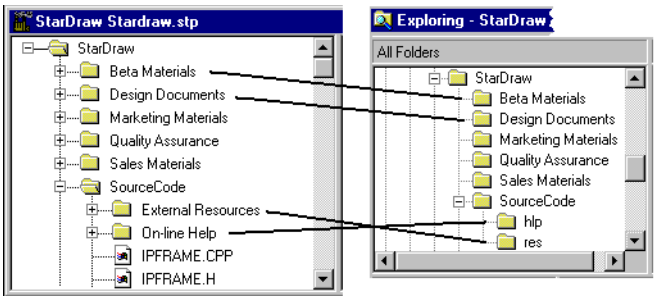
Adding a file puts the initial copy of that file in the vault. That copy can be checked out from the vault by people who need to read it or change it. Before you check out a file to change it, you should lock it. A special icon indicates whether a file is locked. When others see that icon, they know that someone is working on the file and they should not work on it themselves until is becomes unlocked.

NOTE
Versions has a merge feature, but it is always best to work only on files that you personally have locked.

Using Working Files

When you check out a file, Versions copies the desired version of that file from the vault to your workstation's hard drive. This is called the working file and has the filename that appears in the project hierarchy.

To make working files easy to locate, Versions allows administrators to specify directories to store working files on every user's hard drive. The project itself and each subproject has such a directory, called its default working directory. Often the working directories form a hierarchy similar to the project hierarchy. For example, the following figure compares the StarDraw project hierarchy (in a Versions project window) with its folder/directory structure on your hard drive (in an Explorer window).



Notice that these hierarchies are almost identical. The Beta Materials subproject maps to the Beta Materials directory, and the Design Documents subproject maps to the Design Documents directory. Notice the differences: The External Resources subproject maps to the res directory, and the On-line Help subproject maps to the hlp directory. In this case, the only difference is that more descriptive names are given to the subprojects than to their corresponding working directories. Also, because both subprojects and directories are in alphabetical order, the order for External Resources and On-line Help is different than the order for the hlp and res directories.

However, a project hierarchy does not have to map so nicely onto the hard drive. For example, the Source Code subproject could map to C:\Source Code while the Beta Materials subproject could map to C:\External Testing\StarDraw Beta\Beta Materials.

In addition, Versions allows you to specify working directories on a per workstation basis. Depending on what options are set for the project, you can tell Versions to ignore the default working directory and put files from the Beta Materials subproject in C:\StarDraw Beta on the workstation in your office and in D:\Beta Materials on your laptop.

The important things to remember are that the versions are stored in the vault and copied to your hard drive. The location to which they are copied depends on the default working directory specified for the subproject or on the working directory you specify for that subproject on your workstation. You cannot change or even look at a file until a version of it has been copied to your hard drive.

The version you read or change is always the working file on your hard drive. This is why it is so important not to work on files that you have 1) not locked and 2) not checked out the most recent version of. Suppose you checked out the most recent version of a file, but didn't lock it and didn't work on it for several days. Your working file could be two versions older than the most recent one. If you change it and then check it in, the changes made by your coworkers would be missing from the most recent version and lost unless they are retrieved from the earlier versions. To protect your work and that of your coworkers, Versions only allows you to check in files that you have locked. It warns you if your working file is not the most recent version.

Creating New Versions

Checking in a file adds another version of that file to the vault. You check it in (and perhaps unlock it for others):

- When you are done working on it.
- When you have made a significant number of changes and want to save them.
- To be sure that you can go back to the file in its current state later. (Usually before you make some drastic changes that you might not want to keep.)

Even with the space-saving measures that Versions can take, the vault could become very large if you saved every single version of each file. Therefore, Versions allows you to limit the number of versions to be kept. For example, suppose you or your administrator decide to keep only 10 versions of a particular file. Then when the 11^{th} version is added, the first (the oldest) is deleted. Because you might want to save some versions, regardless of their age, Versions allows you to mark any such versions as permanent. Permanent versions are never deleted. Nor are they counted among the 10 versions to be kept, in this example. That means you keep all the permanent versions in addition to the specified number of non-permanent versions.

Using Security

Versions has security features that can allow some users to perform actions that others cannot. Security can be implemented in any of the following ways:

- No security. Any user can perform any action, such as adding subprojects or deleting files.
- Project-level security. Any user with administrator status can perform any action. Users without administrator status cannot perform actions that set project/subproject properties or that can result in the loss of data (such as deleting files and defects). See Chapter 2, “Managing Projects” for more details.
- A combination of project-level and subproject-level security.

  An administrator can create security templates that define sets of actions to be performed. Then users or groups of users can be allowed to perform the actions defined by the template on a subproject-by-subproject basis. Suppose there are two different security templates names Power User and Guest in a project that has three different user groups name Programmers, Testers, and Writers. For the Source Code subproject, the administrator can allow the users in the Programmers group to perform the actions defined by the Power Users template while the users in the Testers and Writers groups can perform only the actions defined by the Guest template. For example, the programmers can add, modify, and delete files from the subproject, while the testers and writers can only read them. However, for the User Manual and On-line Help subprojects, writers would have Power User privileges while programmers and testers would have Guest Privileges. See Chapter 3, “Managing Subprojects” for more details.

Chapter 2
Managing Projects

A project is a way to group related files and place them under version control. For example, if you create a project for software product development, the files containing the product's functional specification, marketing requirements document, source code, and test suites can be stored in subprojects.

Although you may organize projects based on the file directory structure, the logical Versions subproject hierarchy does not have to match your physical disk directory hierarchy.

Versions makes version control easy and convenient.

- Versions keeps track of which files need to be checked in or out so that you don't have to.
- List expansion and multiple or extended selection allow you to check files in or out from several subprojects in a single operation.
- You can use keyword expansion to embed version control information into text files for quick and convenient reference.
- Version control is configurable. For example, when you check out a file you do not have locked, you can require the working copy to be made read-only.

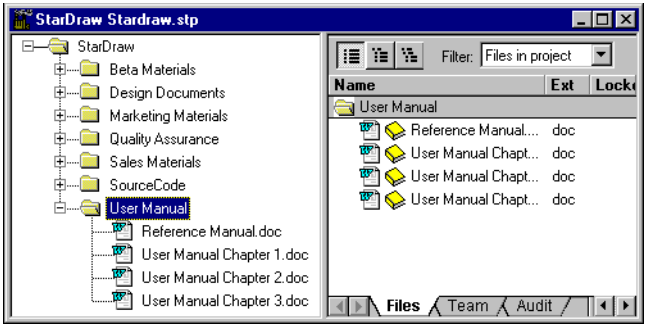
In addition to providing version control for files, you can:

- Limit project access to specific password-protected users.
- Monitor user activity with the audit log. For example, the log indicates when anyone creates or deletes a subproject, purges a file, or adds another team member.
- Quantify the accumulated data using charts and reports.
- Branch subprojects.
- Merge text file changes.

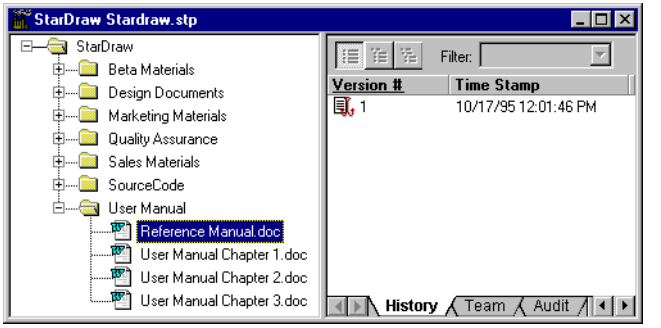
Using the Project Tree

The project tree is the hierarchical display of a Versions project and its associated subprojects and files. The project tree is always displayed in the left pane.

In the project tree shown below, StarDraw is the root subproject. It has the same name as the project. The selected subproject is User Manual. Because User Manual is expanded, its files are listed beneath it in the project tree in the left pane. Because User Manual is also selected, its files are displayed in the files list in the right pane.



In the next figure, the right pane displays the history list, a list of all the versions stored in the vault for the selected file. When a file in the User Manual subproject is selected in the project tree (in the left pane), the Files tab automatically changes to the History tab, the history list replaces the files list, and Versions disables the scope buttons and Filter drop-down list box above the right pane.



Using the Left Pane

Icons

A folder icon precedes each subproject name. An icon identifying the type of file precedes file names. Clicking the name or the icon displays information about that item in the right pane. What kind of information is displayed depends on what object is selected from the project tree and what tab is selected from the right pane.

Expand and collapse

You can expand or collapse branches of the project tree:

A plus sign identifies a collapsed branch. Clicking it expands the branch and changes the closed folder icon to an open folder icon.

A minus sign identifies an expanded branch. Clicking it collapses the branch and changes the open folder icon to a closed folder icon.

Order of subprojects and files

When you expand a subproject, its child subprojects precede the list of its files. Both sets (child subprojects and files) are in ascending alphanumeric order. Within the tree, you can drag a file or subproject to a new parent subproject. For example, a file at the project level can be moved to one of the project's subprojects. However, if security is enabled, only an administrator can move files.

Right-click for context menu

After selecting a subproject or a file from the project tree, use the Project menu or right-click to display a context menu.

Double-click

Double-click file names in the project tree to run executable files or to open files in the application with which they are associated.

Double-click subproject names in the project tree to expand or collapse branches of the tree.

Using the Right Pane

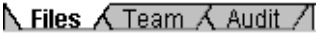
One-to-many

The project tree has a one-to-many relationship with the lists that appear in the right pane. For example, several files in the right pane can be in the subproject selected from the left pane. Many audit entries in the right pane can be associated with the file selected in the left pane. The project tree always appears in the left pane, while the files list, history list, team list, or audit log appears in the right pane.

Control the right pane

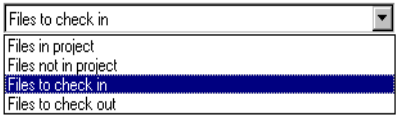
What the right pane displays depends on all of the following:

- The object selected from the project tree.
- The tab selected in the right pane: Files, History, Team, or Audit. A menu, corresponding to the selected tab, becomes the second menu on the menu bar. For example, if you select the Audit tab, the audit log appears in the right pane and the Audit menu appears on the menu bar.



Filtering the pane

- The filter selected from the Filter drop-down list above the right pane. For example, the following figure shows the filter choices available when you select the Files tab.



- The scope button selected above the right pane indicates the depth of the project tree hierarchy for which Versions displays information in the right pane. The leftmost button displays the fewest entries, and the rightmost button displays the most.

	No Descendants	Displays information for the item selected from the project tree only.
	Immediate Descendants	Displays information for the item selected from the project tree and its children only.
	All Descendants	Displays information for the item selected from the project tree, its children, its children's children, and so forth.

As you move from one tab to another, No Descendants becomes the selected scope button. Versions assumes that initially you want to see only one level of data for the new tab.

Sorting

- Clicking a column header sorts the displayed data based on the value in that column. The sort order for most columns is ascending alphanumeric. To change the sort order from ascending to descending (or vice versa), click the header a second time. The primary sort's column header becomes underlined. Right-click another header for a secondary sort.

Group bands

When sorting, group bands separate entries with the same value in the sort column. When the values of the sorted column are unique (such as a time stamp), no group bands appear.

Storing Data

The project database file stores version history and so forth. When you create a project, you provide the project database file name (with the extension .stp) and a directory path for this file.

Projects are accessed locally (via file server access). That means that all team members must have read/write, create, and delete rights to the directory containing the project database file.

The versions of the files you place under version control go into a project vault directory. Versions uses the name of the project database file (with the extension .vlt) to create the vault directory that stores file versions for the project. The vault directory is in the same directory as the project database file.

You can compress or decompress the vault files to save disk space or perhaps speed disk access. See the “Compressing and Decompressing Vault Files” section in Chapter 4 for more information. Do not delete or modify vault files in any way.

Using Working Directories

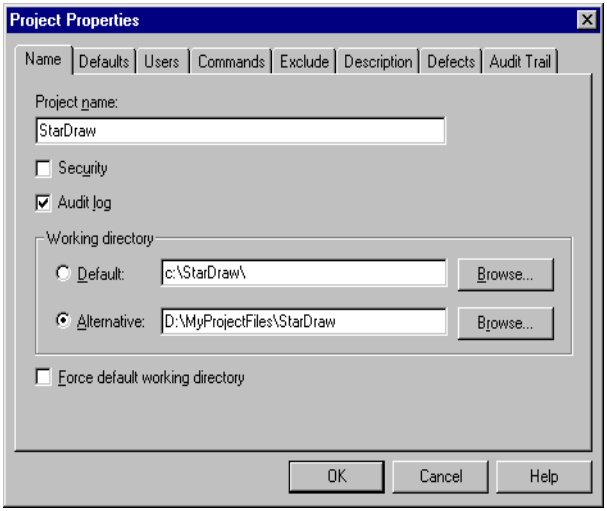
Each project has a working directory, a directory on your workstation from which you read or modify the root subproject’s files. Often a root subproject has only subprojects attached to it, but Versions still requires you to specify a working directory for the root subproject. You specify a project’s default working directory as you create the project. Using the default directories for projects and subprojects means that one team member working on files on one workstation has the same directory structures for those files as another team member working on another workstation. However, depending on the project’s options, you can designate working directories on a per workstation basis. For example, the default working directory for the StarDraw sample project is C:\StarDraw, but you might use D:\StarDraw on the workstation in your office and C:\StarDraw Sample on your laptop.

To change the working directory for the root subproject on your workstation:

1. Select the root subproject from the project tree in the left pane.

2. Select **Project** ⇨ **Properties...** from the menu bar.

   The Project Properties dialog displays the Name property page.



3. Select the Alternative option button.

4. Type the path to the working directory for your workstation.

   Or,

   Click Browse... to simplify this process.

☆ **NOTE:**
When security is enabled, only an administrator can select or clear the Force Default Working Directory check box. When it is selected, users cannot set alternative working directories for that subproject.

Using Security

Security is set at the project level. It can be modified at the subproject level using the security templates explained in Chapter 3, “Managing Subprojects.” At the project level, the actions a team member can perform depend on whether or not the team member is an administrator.

To enable or disable security:

1. Select the root subproject from the project tree.
2. Select **Properties...** from the Project menu or context menu.

   The Project Properties dialog opens
3. Select to enable or clear to disable the Security check box.

The Administrator’s Role

The first team member added to a project automatically becomes its administrator. You add this team member as you create the project.

When project security is enabled and security templates are not used, only administrators can:

- Create subprojects or branch subprojects
- Modify project and subproject properties. For example, this means that only an administrator can enable security.
- Move files or subprojects using drag-and-drop
- Modify file properties
- Delete versions
- Purge non-permanent versions
- Change a version’s status from permanent to non-permanent

- Change the number of non-permanent versions stored for a file
- Delete projects
- Remove subprojects or files from version control
- Manage the team list
  - Add or delete users and groups
  - Give users passwords or administrator status
  - Modify user properties (non-administrators can only view the properties)
  - Add users to or remove them from groups
- Break locks set by other users
- Designate builds and milestones
- Clear the audit log

Depending on the security template used at the subproject level, you can allow some non-administrators to perform file actions (such as deleting files) that are normally restricted to administrators.

The Team Member's Role

Version control can be important for one individual working on a single file that will have several iterations. However, it is crucial when a group of individuals work on the same set of files. Using Versions, team members can lock the files they are currently changing to prevent others from accidentally overwriting those files. After a team member modifies a working file and checks it in, Versions stores the file's contents as a new version in the project vault.

Depending on the security template used at the subproject level, you can restrict the actions of some team members even more. For example, you can prevent some members from checking in files.

When project security is enabled and security templates are not used, any team member (referred to as a user by Versions) can:

- Add files to projects and subprojects
- Lock and unlock files

- Check files in and out
- Merge files
- Change a version's status from non-permanent to permanent
- Change a personal password
- Change the full name, a user property used for e-mail purposes
- View the audit log
- Create reports and charts

Creating a Project

Creating a project allows you to put files under version control and audit users' actions.

Before you create the project, you may need to consider the following:

- Will you need project security, so that actions such as deletions can only be performed by an administrator or specified users?
- Is the project's granularity sufficiently fine? For example, builds and milestones affect every file in the project. If you have subprojects that would normally have their own build and milestone cycles, those subprojects may need to be independent projects with their own project database files (.stp).
- Is the project's granularity excessively fine? For example, if the project has the same build and milestone cycle and shares numerous files with another project, perhaps the two projects should be subprojects of one project instead of independent.

 **TIP**

The first user, added when the project is created, is always an administrator by default. When some users create projects, they log on for the first time using the name Administrator. This makes Administrator a team member. The administrators also make themselves team members using their own names.

When you create a project, you provide a project name, designate a working directory, and provide the name and location of the project database file (.stp).

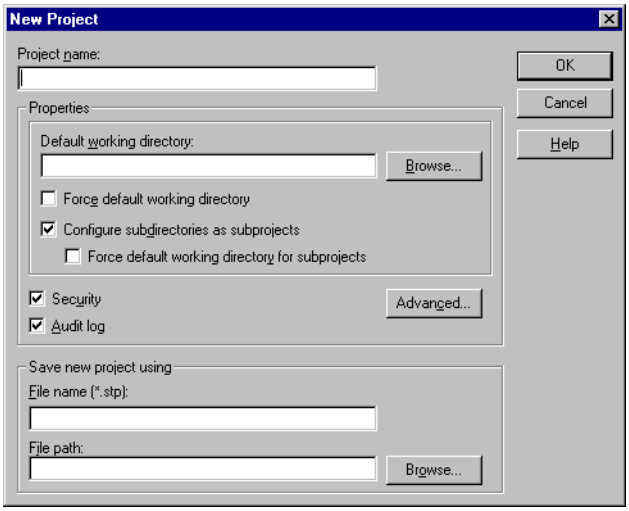
To create a project:

1. Click New Project  .

   Or,

   Select **Project** ⇨ **New...** from the menu bar.

   The New Project dialog opens.



2. Type a name for the project in the Project Name text box. The project name should reflect the nature of the project. Use a maximum of 228 characters. Do not use forward slashes (/), backward slashes (\), or exclamation points (!).

3. Type the path to the directory to be used on individual workstations as the default working directory for this project in the Default Working Directory text box.

   Or,

   Click Browse... to locate that directory.

Do not use Universal Naming Convention (UNC) pathnames or all users will be forced to use the same working directory. For example, the working directory c:\bigproj\working specifies a working directory on each user's local c: drive, while the working directory \\domain\mydisk\working specifies that physical location for a single working directory shared by all team members.

4. Select the Force Default Working Directory check box to require all users to use the root subproject's default working directory for working files on their workstations.

   Clear this box to allow users to specify their own working directories for these files.

5. Select the Configure Subdirectories As Subprojects check box so that all subdirectories of the project's working directory become the default working directories of subprojects. The names of the subdirectories become the names of the subprojects.

   Clear this box to create no subprojects at this time.

6. The Force Default Working Directory For Subprojects check box is enabled only when you select Configure Subdirectories As Subprojects. Select this box to require all users to use each subproject's default working directory for working files on their workstations.

   Clear this box to allow users to specify their own working directories for these subprojects.

7. The Security check box is selected by default, requiring passwords to log on to the project and security privileges to perform certain actions. Clear it to allow anyone complete access to the project.

8. The Audit Log check box is selected by default so that all actions performed on the project will be logged. Clear it to disable logging events.

9. Type name of the project database file in the File Name text box. By default, the file name has the extension .stp.

10. Type the path to the directory where you will store the project database file. Versions also stores the project vault there.

    Or,

    Click Browse... to simplify this process.

Usually the project database file is on a network drive in a location that is accessible to all team members.

11. Click Advanced... to set additional options.

    The Advanced Projects Settings dialog opens.

    Use it to specify:

    - Defaults for the project and for project files

        a. Select the Defaults tab.

            The Defaults property page opens.

            Versions can store a maximum of 32767 versions of files using either delta storage or omega storage.

        b. Select the Maximum or Limited option button for one of the storage types.

        c. If you selected Limited, type a number of versions in the Limited text box.

        d. Repeat steps b and c for the other type of storage, if applicable.

        e. To stamp working files with the time the version is copied to the working directory, select Time Stamp Working File With Check-Out Time.

            Clear this box to copy the time stamp of the version along with the version.

        f. Select the Use Mail check box to use a MAPI-compliant mail system from Versions. For example, if you are logged on to e-mail, others are notified if you break their locks on files.

        g. Select the Keyword Expansion check box to use keywords in your text files. The keywords will be replaced with information stored by Versions. For example, $Author$ is replaced with the name of the user who checked in the version. See the “Using Keywords in Files” section of Chapter 4 for details.

        h. Select the Use Compression check box to compress all the files stored in the vault.

    - Build and test commands:

        a. Select the Commands tab.

        b. Type the path to the command executable or batch file in the Build Command and/or Test Command text box.

Or,

Click Browse to locate the executable or batch file within your system environment and enter its name into the text box automatically.

- Files never to be added to the root subproject

  a. Select the Exclude tab.

  b. Select the Use Exclude List check box.

  c. Type the file specifications in the Exclude List text box. Separate the specifications with semicolons or commas. For example, use "*.obj, *.bak" and Versions ignores all the *.obj and *.bak files in the working directory. They never appear in the list of files that are not in the project.

- A project description:

  a. Select the Description tab.

  b. Type a project description of up to 255 characters.

12. Click OK.

    The New User dialog opens.

To finish creating a project:

1. In the User Name text box, type the name you will use to log on to this project or the name Administrator (to create a first user that can be used by anyone performing administrative functions).

   Select this name carefully, because it cannot be changed. (Usually people use their network logon name.)

2. In the Full Name text box, type your e-mail name if you are using e-mail within Versions.

   When using Microsoft Exchange, the full name and the Exchange profile name should be the same. Otherwise Microsoft Exchange returns an error message each time you log on to the Versions project. If this occurs, create a new Exchange profile with the same name as your Versions full name.

3. If you enabled project security, you must also type and confirm a password.

   Passwords are case-sensitive and must contain at least six characters.

4. Click OK again.

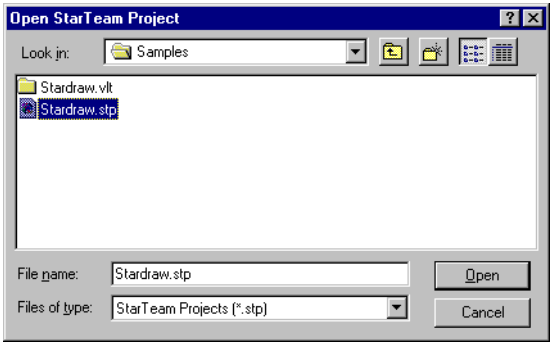
Accessing an Existing Project

You access a project by opening it and logging on to it.

To log on to a project:

1. Select **Project** ⇨ **Open...** from the menu bar.

   The Open dialog opens.



2. Locate and select the project database file (.stp) for your project, then click Open.

   The Log On dialog opens.

3. Type your logon name in the User Name text box.

   If security is disabled, User Name is the only text box in this dialog. If you are not already a team member for a project without security enabled, logging on adds you to the team with the name you have specified. Select your name carefully, because it cannot be changed.

The full name and the Exchange profile name must be the same. Otherwise Microsoft Exchange generates an error message each time you log on to the Versions project.

4. If security is enabled, type your password in the Password text box.

   Your password is blank if you used this project without a password at a time when security was not enabled yet.

5. Click OK.

   The Project window displays the tree for your project. The name of your project and its .stp file appears on the title bar.

   By default, the item selected from the tree is the root subproject. It has the same name as your project.

TIP
Select the name of a recently opened project from the bottom of the Project menu.

Modifying Properties

As the project administrator, you can modify the following project properties:

- Name
- Working directory
- Use of:
  - Audit log
  - Compression
  - Mail
  - Security
  - Keyword expansion
  - Check-in features
- Description

- Maximum versions
- User and group privileges to the files in the root subproject
  (See the “Setting Up Subproject Level Security” section in Chapter 3 for details.)
- Build and test commands
- List of excluded file specifications
  (See the “Exclude Files” section in Chapter 3 for details.)
- Time stamp for working files

You can also review the audit log assigned to the project from the Project Properties dialog. The Audit Trail property page has a context menu that allows you to perform operations.

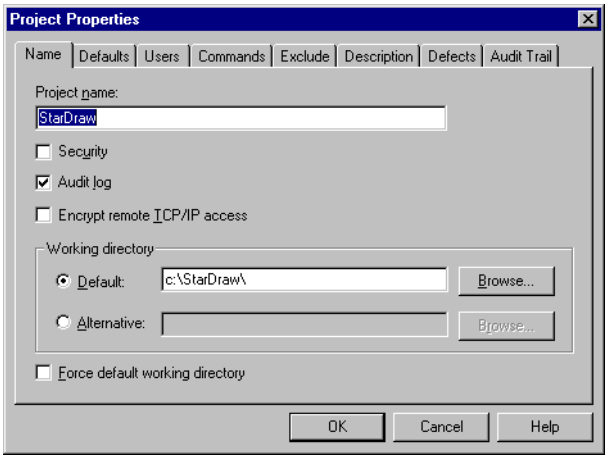
For more detailed information about modifying these properties, see “Creating a Project” earlier in this chapter. The properties were originally set when you created the project. For more information about assigning privileges to users via the Users tab, see the “Setting Up Subproject Security” section in Chapter 3.

To modify project properties:

1. Select the root subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

The Project Properties dialog opens.



3. Select the appropriate tab: Name, Users, Defaults, Commands, Exclude, Description, Defects, or Audit Trail.

4. Change the settings for the properties.

Building and Testing Projects

Versions allows you to associate a Build and/or Test command with the root subproject. Once the commands are associated with the root subproject, command execution is only a menu selection or button click away.

To execute a Build command:

1. Select the root subproject.
2. Select **Project** ⇨ **Build** from the menu bar.

   Or,

   Click Build  .

To execute a Test command:

1. Select the root subproject.
2. Select **Project** ⇨ **Test** from the menu bar.

   Or,

   Click Test  .

Designating a Build

A build represents a quantifiable step of progress. For example, a software development project may deliver a new build every few days to the team members who test the product. This feature is not available at the subproject level. When project security is enabled, only an administrator can designate a build.

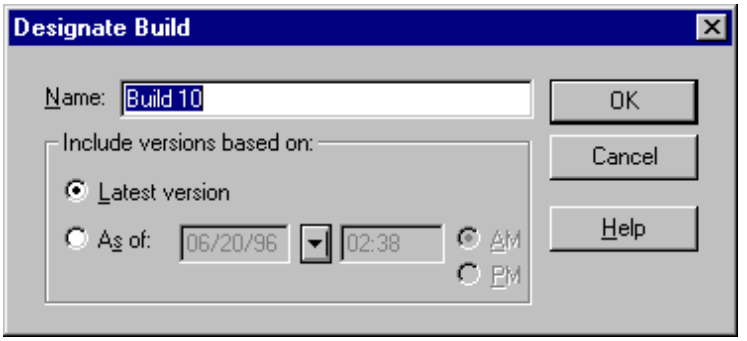
When you designate a build, you create a single set of versions for the files in the project. The specified version of each file receives the build label.

For example, suppose the same version of a file ends up in two separate builds and a milestone. That version stores both build labels and the milestone label. A version can have any number of labels.

To designate a build:

1. Select the root subproject from the project tree.
2. Select **Project** ⇨ **Designate Build...** from the menu bar.

   The Designate Build dialog opens.



3. Type a name or label for the build in the Name text box.
4. Indicate what versions to include by selecting one of the following option buttons:
   - Latest Version, for the most recent versions.
   - As Of, for the versions checked-in at or before the date and time you specify.
     a. To specify the date, type the date in the text box.
        Or,
        Click the button between the date and the time to use the calendar. Specify the date using the next and previous month buttons and the arrow keys.
     b. To specify the time, type the time in the text box, then select the AM or PM option button.

Designating a Milestone

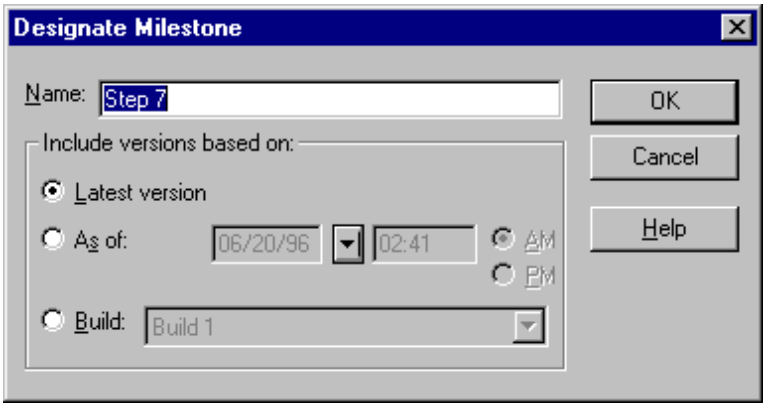
A milestone represents a significant step in progress for a project. For example, the label may indicate the alpha, beta, or final release of a product. When project security is enabled, only an administrator can designate a milestone.

In Versions, a milestone designates a single set of versions for the files in a project. The specified version of each file receives the milestone label and becomes permanent.

To designate a milestone:

1. Select the root subproject from the project tree.
2. Select **Project** ⇨ **Designate Milestone...** from the menu bar.

   The Designate Milestone dialog opens.



3. Type a name or label for the milestone in the Name text box.
4. Indicate what versions to include by selecting one of the following option buttons.
   - Latest Version, for the most recent versions.
   - As Of, for the versions checked-in at or before the date and time you specify.
     a. To specify the date, type the date in the text box.
        Or,
        Click the button between the date and the time to use the calendar. Specify

the date using the next and previous month buttons and the arrow keys.

b. To specify the time, type the time in the text box, then select the AM or PM option button.

- Build, for the versions in the build specified using the Build drop-down list box.

Changing Project Defaults

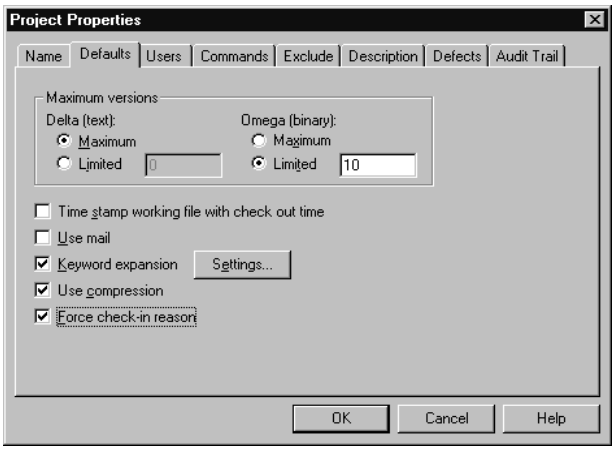
You can change the project defaults that control how many versions of files are stored in the vault, how checked-out files are time stamped, whether keyword expansion and compression are used, and whether mail is enabled.

To change project defaults for the project:

1. Select the root subproject from the project tree.
2. Select **Properties...** from the Project menu or context menu.

   The Project Properties dialog opens
3. Select the Defaults tab.

The Defaults property page opens.



The number of non-permanent versions stored for any file in the project depends on the file's storage type. By default, Versions stores a maximum of 32767 versions of files using either delta storage or omega storage.

4. Select the Maximum or Limited option button for one of the storage types.

5. If you selected Limited, type a maximum number of versions in the Limited text box.

6. Repeat steps 4 and 5 for the other type of storage, if applicable.

7. To stamp working files with the time the version is copied to the working directory, select Time Stamp Working File with Check-Out Time.

   Clear this box to copy the time stamp of the version along with the version.

8. Select the Use Mail check box to use a MAPI-compliant mail system from within Versions. For example, if you are logged on to e-mail, others are notified if you break their locks on files.

9. Select the Keyword Expansion check box to use keywords in your text files. The keywords will be replaced with information stored by Versions. For example, $Author$ is replaced with the name of the user who checked in the version. See the "Using Keywords in Files" section of Chapter 4 for details.

10. Select the Use Compression check box to compress all the files stored in the vault.

11. Select Force check-in reason to require users to enter check-in comments.

Moving a Project and Project Vault

Versions does not automate the process of moving a project. You can, if the need arises, move the project manually. Be sure to move both the project database file and the project vault directory to the new location.

To move a project and project vault:

1. Before you begin, do both of the following:
   - Verify the name of the project database (.stp) file, which appears in the title bar of the project window, then close the project.
   - Verify that no team members are currently logged on to the project.

2. Using File Manager or Windows Explorer, locate the .stp file to be moved. In the same directory, locate the .vlt directory that stores the vault files for the project database file. (For example, if the project file is named project1.stp, the project vault directory is named project1.vlt.)

3. Using any valid method, move the .stp file and the .vlt directory to the new location.

4. Notify all team members of the new project location.

If you fail to move the vault and attempt to check out a file, Versions cannot find it and returns the error "The vault file does not exist."

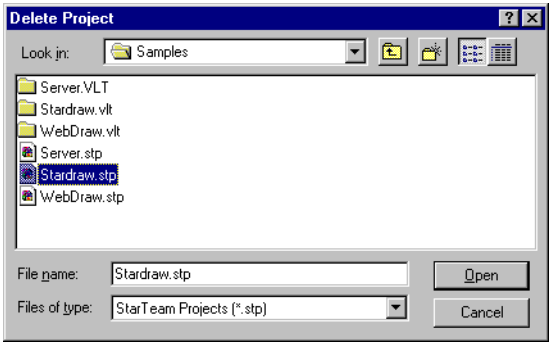
Deleting Projects

Deleting a project is extremely dangerous. It deletes all project data forever. When project security is enabled, only an administrator can delete a project. Use the following procedure with great caution.

To delete an entire Versions project:

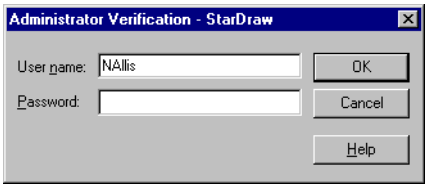
1. Before you begin, do both of the following:
   - Verify the name of the project database (.stp) file, which appears in the title bar of the project window, then close the project.
   - Verify that no team members are currently logged on to the project. Versions cannot delete a project that is in use.
2. Select **Project** ⇨ **Delete...** from the menu bar.

   The Delete Project dialog opens.



3. Locate and select the project database (.stp) file to delete.

The Administrator Verification dialog opens.



4. Confirm project deletion by typing your password.

   Versions deletes the project database file, the project vault, and all stored file versions. Working copies of files stored locally are not deleted.

Chapter 3
Managing Subprojects

Subprojects can be created automatically at the same time you create a project or a parent subproject or they can be added later. If they are created automatically, all of the following are true:

- Their default working directories are subdirectories of the default working directory for the project or parent subproject.
- Their names are the same as the names of the subdirectories.
- You can make the decision about whether their default working directories are mandatory as you create the project or parent subproject.

The root subproject, which has the same name as the project, is always created at the same time as the project. To change the project properties, this subproject should be selected.

If subprojects are created later:

- Their default working directories can be any directories you choose.
- Their names can be different than the names of their working directories.
- You can make the decision about whether the default working directories are mandatory as you create the subproject.

For more information on working directories, see “Using Working Directories” later in this chapter.

Managing a subproject is very similar to managing a project. You can create and delete them, and modify their properties. Their properties are, for the most part, a subset of the properties for a project.

The main difference is that you can assign privileges on a subproject-by-subproject basis to individual users or groups of users by means of security templates. Security templates are defined at the project level and applied at the subproject level.

Using Working Directories

You use a directory, called a working directory, on your workstation to edit or modify files that belong to a subproject. Each subproject has a default working directory. Using the default directory means that one team member working on subproject files on one workstation has the same directory structure for those files as another team member working on another workstation. However, depending on the subproject's options, you can designate a working directory on a per workstation basis. For example, the default working directory might be "C:\StarDraw\Beta Materials", but you might use "C:\StarDraw Beta" on the workstation in your office and "D:\StarDraw Beta" on your laptop.

Versions expects you to add and check in subproject files from a working directory. When you check out subproject files, Versions copies files versions to the subproject's working directory. If the specified working directory does not exist, Versions creates it for you as you check out the files.

You can also select **Project ⇨ Subproject ⇨ Create Directory Structure** from the menu bar to create the needed directory structures. For example, if you want to add a file to an existing subproject, but do not need to check out any files, use Create Directory Structure, move the file to the working directory, then add it to the project.

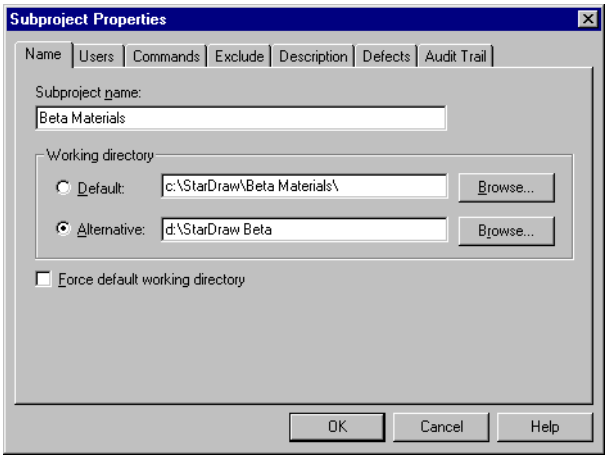
If you create a subproject for which a directory structure already exists on your workstation, you can request that Versions make a subproject for each subdirectory in the structure.

To change the working directory on your workstation for a subproject:

1. Select a subproject from the project tree in the left pane.

2. Select **Project ⇨ Properties...** from the menu bar.

The Subproject Properties dialog displays the Name property page.



3. Select the Alternative working directory option button.

4. Type the path to the working directory for your workstation.

   Or,

   Click Browse... to simplify this process.

NOTE:
When security is enabled, only an administrator can select or clear the Force Default Working Directory check box. When it is selected, users cannot set alternative working directories for that subproject.

Creating a Subproject

Creating a subproject allows you to put files under version control and audit users' actions.

When project security is enabled, only an administrator can create a subproject.

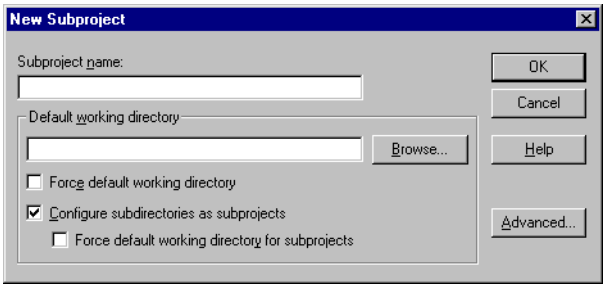
To create a subproject:

1. From the project tree, select the subproject to which the new subproject will be long.
2. Click New Subproject  .

   Or,

   Select **Project** ⇨ **Subproject** ⇨ **New...** from the menu bar or **New Subproject...** from the context menu.

   The New Subproject dialog opens.



3. Type a name for the subproject in the Subproject Name text box. Use a maximum of 228 characters. Do not use forward slashes (/), backward slashes (\), or exclamation points (!).
4. Type the path to the default working directory for the new subproject.

   Or,

   Click Browse... to locate that directory.

   If the working directory does not exist, Versions can create it for you.

Do not use Universal Naming Convention (UNC) pathnames or all users will be forced to use the same working directory (instead of local directories with identical pathnames). For example, the working directory c:\bigproj\working specifies a working directory on each user's local c: drive, while the working directory \\domain\mydisk\working specifies that physical location for a single working directory shared by all team members.

5. Select the Force Default Working Directory check box to require each user to store the working copies of the subproject files in the working directory on his or her workstation with the path that you specified as the default.

   Clear this box to allow users to specify their own working directories for these files.

6. Select the Configure Subdirectories As Subprojects check box so that all subdirectories of this subproject's working directory become the working directories of subprojects. The names of the subdirectories become the names of the subprojects.

   Clear this box to create no additional subprojects at this time.

7. The Force Default Working Directory For Subprojects check box is enabled only when you select Configure Subdirectories As Subprojects. Select this box to require every user to store each subproject's working files on his or her workstation in the working directory specified by the default working directory and the path from it to the subdirectory.

   Clear this box to allow users to specify their own working directories for these files.

8. Click Advanced... to set additional options.

   The Advanced Subproject Settings dialog opens.

   Use it to specify:

   - Privileges for users and groups.

     See the "Setting Up Subproject Level Security" section later in this chapter for more information.

- Build and test commands:

  a. Select the Commands tab.

  b. Type the path to the command executable or batch file in the Build Command and/or Test Command text box.

     Or,

     Click Browse to locate the executable or batch file within your system environment and enter its name into the text box automatically.

- Files never to be added to the subproject

  a. Select the Exclude tab.

  b. Select the Use Exclude List check box.

  c. Type the file specifications in the Exclude List text box. Separate the specifications with semicolons or commas. For example, use "*.obj, *.bak" and Versions ignores all the *.obj and *.bak files in the working directory. They never appear in the list of files that are not in the subproject.

- A subproject description:

  a. Select the Description tab.

  b. Type a subproject description of up to 255 characters.

To finish creating the subproject:

1. After completing the steps explained above, click OK.

   The new, but empty, subproject appears in the project tree. Versions adds the new subproject to the project database.

2. Select the Files Not In Project filter from the Filter drop-down list box to see the files that can be added to the subproject and, therefore, to the project vault.

Setting Up Subproject Level Security

To setup subproject level security, you must create security templates and then assign users or groups to those templates on a subproject-by-subproject basis. While you can create templates and assign users and groups under any circumstances, security is not enforced unless it is set as a project property. Administrators always have full privileges in any subproject. The templates apply only to non-administrators.

To enforce security:

1. Select the root subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

3. Select the Security check box.

Managing Security via Security Templates

Use security templates to assign users or groups of users certain privileges normally reserved for administrators. You can also use templates to limit the privileges of some users. Only an administrator can assign security templates to subprojects and assign users or groups to those templates. Each user, group, and template must have a unique name.

A default template, named Default, comes with Versions. It is preset to the usual non-administrator privileges, but it is editable.

Also, a default group of users called All Users comes with Versions. Assigning All Users to a template in a subproject, gives every non-administrator the same privileges in that subproject.

If you assign All Users to the Default security template in the root subproject when no other security templates are used in any subprojects, every non-administrator has the same privileges in every subproject. In fact, that is the default when a new project is created with security enabled.

You can give users or deny them the right to:

- Add files.
- Read files.

  This permits users to check out files (but not check them in).
- Read and write files.

  This permits users to check them out to be changed, and check them back in.
- Delete files.

If no security template is assigned to a user in a subproject, she or he has the right in that subproject that they have in its parent subproject.

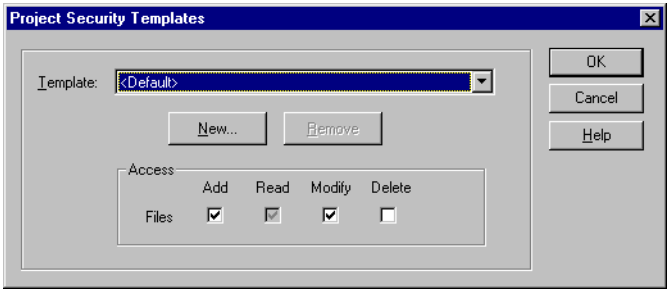
No user or group can be assigned to more than one template per subproject. However, if a user is assigned individually and then in one or more groups in a subproject, she has the maximum (or union) of those privileges in that subproject. For example, if one template allows her to delete files but the others do not, she can delete files.

Deleting users from a project automatically deletes them from any groups and templates to which they were assigned.

To create a template:

1. Select **Project** ⇨ **Security Templates...** from the menu bar.

   The Project Security Templates dialog opens.



2. Click New....

   The Template dialog opens.

3. Type the name for the new template then click OK.

   For example, you might want templates for superusers, general users, and guests.

   The new template name appears in the Project Security Template dialog.

   By default, the first three check boxes are selected, giving users the usual non-administrator rights. The Read check box is always selected and cannot be cleared whenever Modify is selected.

4. Leave the Add check box selected if the users assigned to this template should be allowed to add files to a to-be-determined subproject. Clear it to prevent these users from adding files.

5. Leave the Modify check box selected to allow users to check files in and out. Clear it to prevent these privileges.

6. If you clear the Modify check box, you can also clear the Read check box, preventing users from checking out files.

7. Select the Delete check box to allow users to delete files. Leave it cleared to prevent these users from deleting files.

To modify a security template:

1. Select **Project** ⇨ **Security Templates...** from the menu bar.

   The Project Security Templates dialog opens.

2. Select the security template to be modified.

3. Select and/or clear the appropriate check boxes.

To delete a security template:

1. Select **Project** ⇨ **Security Templates...** from the menu bar.

   The Project Security Templates dialog opens.

2. Select the security template to be deleted.

3. Click Remove.

   The template is deleted. It can no longer be used by any subproject.

Managing Users Privileges Within Subprojects

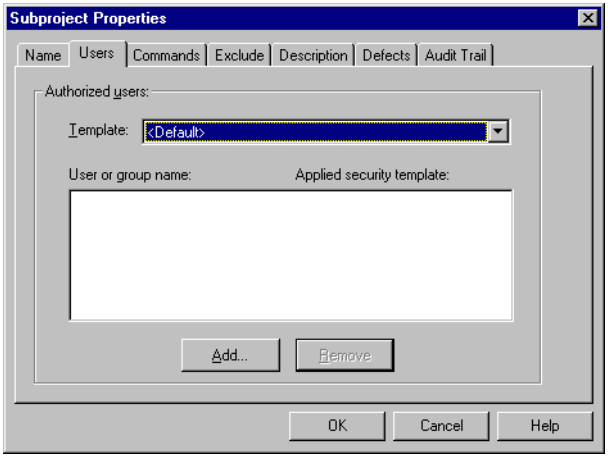
After you create security templates, you can assign users and/or groups to those templates on a subproject-by-subproject basis. In this way you can define a standard set of privileges to be used by different users in different subprojects. For example, the group testers can have superuser privileges only in the Test Suites subproject while the marketing group has superuser privileges only in the Marketing Materials subproject.

To assign users or groups of privileges within a subproject:

1. Select a subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

   The Project Properties (for the root subproject) or the Subproject Properties dialog opens.



3. Select the Users tab.

4. Select a security template from the Template drop-down list box.

   In this way, you select a standard set of privileges.

5. Click Add....

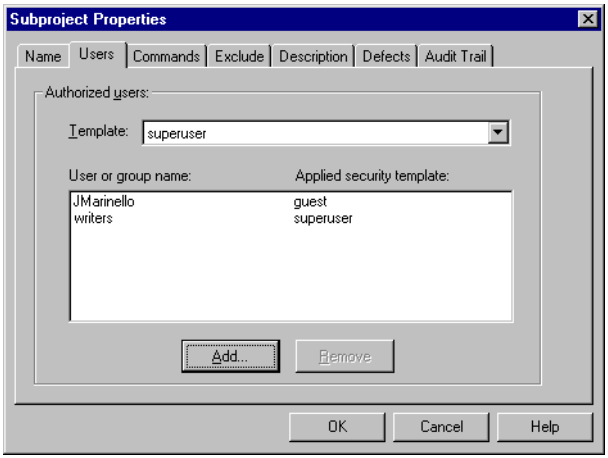
   The Add Users dialog opens.

6. Select one or more users or groups from the list box, then click OK.

   In this way, you determine to whom that standard set of privileges will apply.

7. Repeat steps 4 through 6 for any other templates to be used within this subproject.

   The end result for the User Manual subproject (in the StarDraw project) might be as shown in the following figure:



   This gives JMarinello guest privileges and the members of the writers group superuser privileges within this subproject.

To remove privileges from users or groups within a subproject:

1. Select a subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

   The Project Properties (for the root subproject) or the Subproject Properties dialog opens.

3. Select the Users tab.

4. Select the names of one or more users or groups from the list box.

5. Click Remove.

Modifying Properties

As a project administrator, you can modify the following subproject properties:

- Name
- Working directory
- Description
- User and group privileges to the files in a subproject
- Build and test commands
  (See the “Building and Testing Subprojects” section later in this chapter.)
- List of excluded file specifications
  (See the “Excluding Types of Files” section later in this chapter.)

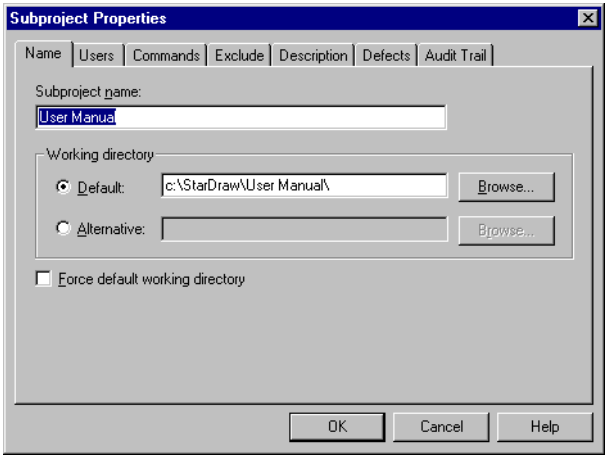
You can also review the audit log assigned to the subproject from the Subproject Properties dialog. The Defects and Audit Trail property page has a context menu that allows you to perform operations. For example, you can review an event’s properties.

To modify subproject properties:

1. Select a subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

The Subproject Properties dialog opens.



3. Select the appropriate tab: Name, Users, Commands, Exclude, Description, Defects, or Audit Trail. (A subproject does not have a Defaults tab. Its security, use of the audit log, etc. are all project-dependent.)

4. Change the settings for the properties.

Building and Testing Subprojects

Versions allows you to associate a Build and/or Test command with each subproject. Once the commands are associated with the subproject, command execution is only a menu selection or button click away.

To associate a Build and/or Test command with a subproject:

1. Select the subproject.

2. Select **Project** ⇨ **Properties...** from the menu bar.

   The Subproject Properties dialog opens.

3. Select the Commands tab.

4. Type the path to the command executable or batch file in the Build Command and/or Test Command text box.

   Or,

   Click Browse to locate the executable or batch file within your system environment and enter its name into the text box automatically.

To execute a Build command:

1. Select the subproject to build.

2. Select **Project** ⇨ **Build** from the menu bar.

   Or,

   Click Build  .

To execute a Test command:

1. Select the subproject to test.

2. Select **Project** ⇨ **Test** from the menu bar.

   Or,

   Click Test  .

Excluding Types of Files

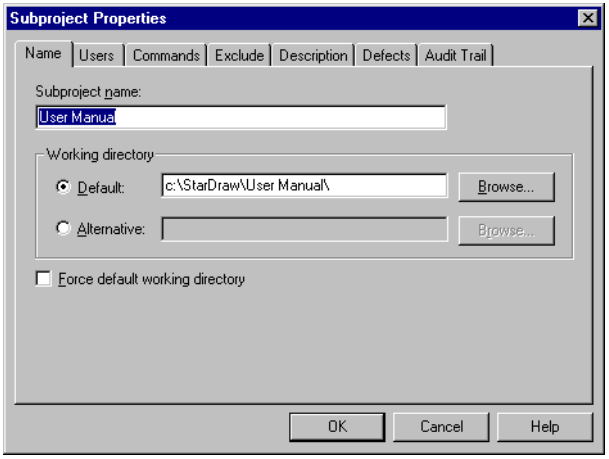
You can exclude certain file types. Users do not see excluded files when they use the Files Not In Project filter so they cannot add them to the subproject. For example, if you never want to add files from a subproject's working directory that have the extension .obj, exclude them. Each exclude list applies to files in only one working directory (the one for the selected subproject).

To exclude file types from view:

1. Select the subproject from the project tree.

2. Select **Properties...** from the Project menu or context menu.

   The Project Properties or the Subproject Properties dialog opens.



3. Select the Exclude tab.

4. Select the Use Exclude List check box.

5. Type the file specifications in the Exclude List text box. Separate the specifications with semicolons or commas. For example, use “*.obj, *.bak” and Versions ignores all the *.obj and *.bak files in the working directory. They never appear in the list of files that are not in the project.

Branching and Merging Subprojects

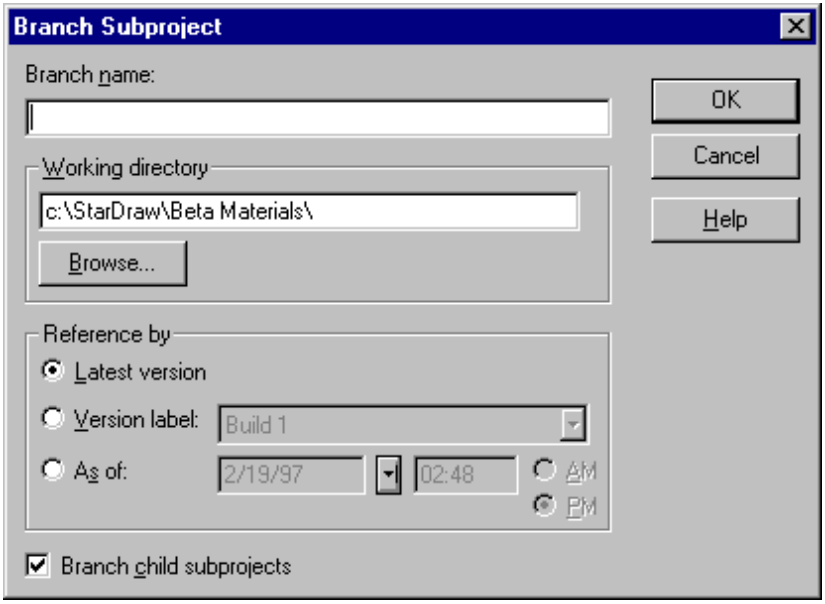
When you branch a subproject, Versions stores versions for the branched files separately from the original file versions. This allows parallel development using two sets of subproject files. Files in subprojects that have been branched can be merged back together at a later date.

To branch a subproject:

1. Select the subproject to be branched.

2. Select **Project** ⇨ **Subproject** ⇨ **Branch...** from the menu bar.

   The Branch Subproject dialog opens.



3. Type a name for the branch subproject in the Branch Name text box.

4. Type the path for the working directory for the branch subproject's files.

   Or,

   Click Browse... to locate the working directory.

   NOTE

   Do not use Universal Naming Convention (UNC) pathnames or all users will be forced to use the same working directory (instead of local directories with identical pathnames). If you specify a directory that does not exit,Versions can create the directory for you.

5. Select one of the following check boxes to indicate what versions will be the initial versions in the branched subproject:

   • Latest version to use the most recent versions.

   • Version Label to use the versions with a specific label selected from the Version label drop-down list box.

   • As Of to use the version checked-in at or before the date and item you specify.

   a To specify the date, type the date in the text box.

     Or,

     Click the button between the date and the time to use the calendar.

   b To specify the time, type the time in the text box, then select the AM or PM option button.

6. To branch the selected subproject and its children based on the above criteria, select the Branch Child Subprojects check box.

NOTE

This check box will only be available if all children subprojects have working directories that are subdirectories of the selected subprojects.

Versions can merge branched files back into the original subproject's working directory. This merge operation incorporates the changes that have been made to the branch file and the changes that have been made to the original file by comparing each of these files with the version at which the branching took place.

To merge the original and branch subproject files, use the Versions command-line merge function (stcmd mrg)—not **Files ⇨ Merge...**. See Chapter 13, "Using the Command-line Interface," for more information on the command-line interface.

Deleting Subprojects

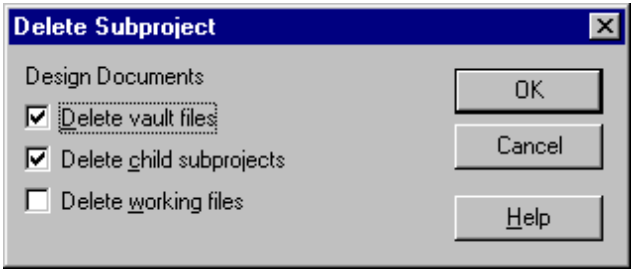
Deleting a subproject from a project deletes all the audit entries and other information associated with the subproject. Optionally, you can delete the vault files, child subprojects, and even working files.

To delete a subproject:

1. Select a subproject from the project tree.

2. Select **Project ⇨ Subproject ⇨ Delete...** from the menu bar.

   The Delete Subproject dialog opens.



3. Select the Delete Vault Files check box to delete the versions in the vault for this subproject. When cleared, these files become attached to the deleted subproject's parent.

4. Select the Delete Child Subprojects check box to delete the subproject's subprojects. When cleared, these subprojects become the children of the deleted subproject's parent.

5. Select the Delete Working Files check box to delete the corresponding working files from their working directories. Clear the box to leave the working files undisturbed.

Chapter 4
Managing Files

When a file is under version control, Versions maintains the information needed to recreate any stored version of that file.

Storage method

Versions uses both omega and delta storage depending on the type of file.

Binary files

Versions always stores each version of a binary file in its entirety (omega storage).

Text (ASCII) files

By default, Versions stores the latest version of a text file in its entirety. For all other stored versions, only the differences between one version and the next are saved (reverse delta storage).

For very large text files that have numerous changes, specifying omega storage may shorten the check-in time because there is no need to check for differences. Once a text file has been stored using omega storage, it will always be stored using omega storage.

Compression

With either method, the versions stored in the vault can be compressed.

How many versions are stored?

Versions stores all versions of the file you designate as permanent. In addition, it stores a specific number of the most recent non-permanent versions. For example, suppose the

maximum is 10 non-permanent versions. When the 11th non-permanent version is checked in, the first non-permanent version is automatically deleted.

You set a default maximum for non-permanent versions of any project file as a property at the project level, but you can override that maximum for specific files as you add them to the project or, later, by changing their file properties.

You can perform additional clean-up by deleting permanent versions or by purging all non-permanent versions from the file. However, one version must always be left in the vault file. To get rid of all versions, delete the file from the project.

Grouping versions of files

Versions allows you to label a set of versions as a build or a milestone. For example, if the project reaches a particular milestone (such as beta) on a particular date, selecting that project and designating a milestone gives the most recent version of each project file the same label. This allows you to check out those versions as a group at any time. Designating a milestone also makes those versions permanent—although designating a build does not.

The same version of a file may end up in several builds or milestones. This is not a problem because a version can have any number of labels. However, no label can be repeated. For example, you cannot add a version to a build or milestone by giving it the same label as that build or milestone.

Labeling versions as you check files in

You can also label a group of versions (or a single version) as you check files in. This makes it easier to check those specific versions out again at a later time.

History and audit log

Versions preserves historical information about each version and logs information about actions performed. For example, the audit log indicates when and by whom a file is added to a project, checked in, and purged of non-permanent versions.

Links to defects

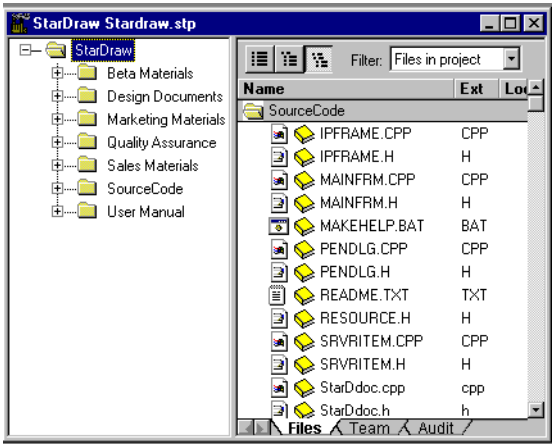
If you use Versions with StarTeam Workstation, you can view and edit defects found in your program. As you check a file in, you can resolve defects associated with it at the same time. Defect resolutions are associated automatically with the next build number. This lets testers know which build to test.

Using the Files List

When you select a subproject from the project tree, the Files tab becomes available. If you select the tab, the files associated with the subproject appear in the right pane, and the Files menu appears on the menu bar.

TIP
Right-click to see context menus.

The files list in the following figure displays all the files in the StarDraw project (sorted by subproject).



What the files list displays depends on:

- What subproject you select from the project tree
- The filter you select from the Filter drop-down list box
- The depth you specify with a scope button

Icons

Two icons precede the name of each file. One indicates the type of file, such as a document, text, or graphics file. The other indicates whether the file is:

 Not locked by anyone

 Locked by you

 Locked by another team member

 Not in the project

Columns

The files list contains several columns, each with its own column header.

The columns for the files list are:

Name	Name of the file (including any extension).
Ext	Extension of the file.
Locker	Person who currently has the file locked.
Exists	Indicates whether the file is in your working directory.
Size	Size of the working file in bytes.
Time stamp	Time stamp for the working file.
Max vers	Maximum number of non-permanent versions to be stored for this file.
Description	Description of the file provided when it was added to the project.
Author	Person who added the file to the project.
Subproject	Subproject containing the file.
Path	Path to the file's working directory.

Sorting

Click a column header to sort the displayed files based on the value in that column. The sort order for each of these columns is ascending alphanumeric. To change the sort order from ascending to descending, click the header a second time. Right-click another header for a secondary sort. The primary sort column's header is underlined.

Using File Filters

Selecting a filter from the Filter drop-down list box allows you to limit the kinds and quantity of files that appear in the files list. What is displayed is also affected by your selection from the project tree and the scope buttons.

The filters are:

Files in project	Displays the files in the working directory that exist in the project.
Files not in project	Displays only the files that are in the project's working directory but not in the project. Unless you add them to the project, their names never appear on the same list as the files that are in your project.
Files to check in	Displays the files in the working directory for which the working file is more recent than the latest checked-in version. Versions checks both the size and time stamp for changes.
Files to check out	Displays the files for which the latest checked-in version is more recent than the working file or for which there is no working file. (Files that are not in the working directory have the value No in their Exists columns.) Versions records the version number for the version of each file currently in your working directory. If that number is less than the current version number for the file, the file needs to be checked out. If, for some reason, no version number is stored, Versions uses the size and time stamp to determine what needs to be checked out.

Using a File under Version Control

Adding a file

To place a file under version control, add it to a project. The file must be in the working directory for either the project or one of its subprojects. Adding the file stores a copy of the working file (the version of the file in the working directory) as the first version in the vault. This means that you can delete the working file without losing any data. Versions recreates that version for you any time you check it out.

Checking out a version

Checking out a version copies that version of the file to a working directory.

Check out a version to be sure that you have the latest or a specific version to work on. For example, you may need the changes made to the file by a team member or you may have deleted the working file earlier, just to save space on your hard drive.

Lock file before changing it

If you intend to make changes to the file, you should lock it as part of the check-out procedure. The lock makes your intention clear to others. Besides, the file must be locked before you can check it back in.

Workstation options for check-in, check-out

The defaults for your workstation options indicate how you want to check files in and out. However, you can override these defaults for specific files by selecting **Files** ⇨ **Check In...** or **Files** ⇨ **Check Out...** from the menu bar.

For example, if you usually change the contents of project files, enable the workstation option Lock Files on Check-out. Then to check out the occasional file that you won't be editing, select **Files** ⇨ **Check Out...** from the menu bar to disable that option. Because the file is unlocked, the

working copy becomes read-only if you have selected the Mark Unlocked Working Files Read-only workstation option.

Checking in a version

After you change a working file, check it in. You may be finished with the file or want to retrieve this version as-is if future changes are cut from the product. Versions records the changes as a new version. As part of the check-in process, you can remove the lock making the file available to others or maintain the lock because you will continue to make changes to the working file. You can also delete the working file.

Check In and Check Out dialogs

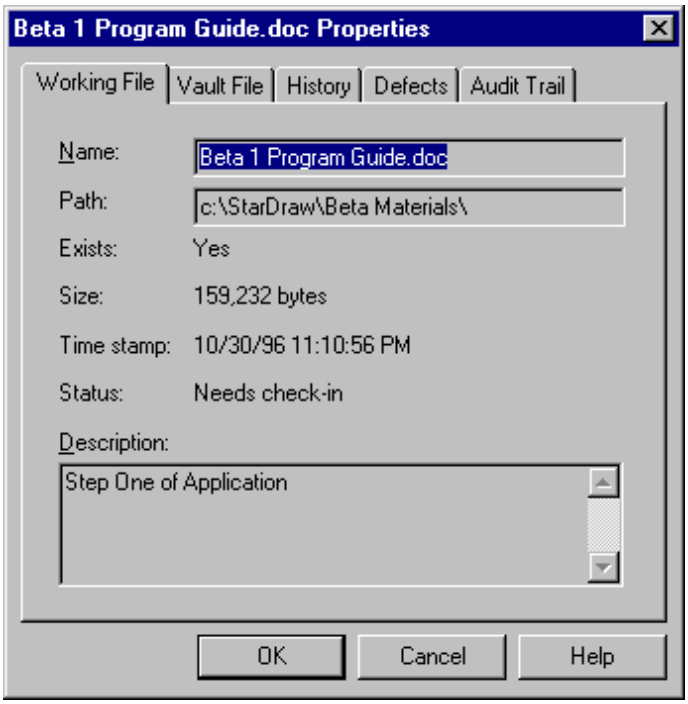
The context menus and toolbar buttons provide quick check in and check out. Only the Check In and Check Out items on the Files menu display dialogs. From the Check In dialog, you can change workstation defaults, provide reasons for checking in files, and resolve defects. From the Check Out dialog, you can check out files based on their version labels or their time stamps. Usually, you check in files using **Files ⇨ Check In...** from the menu bar to document the reason for check in. However, most people do quick check outs because they want only the most recent versions.

Merging files

Mistakes can happen despite the availability of locks and communication tools. If two team members change the same text file simultaneously or if one member changes an outdated file, you can use Versions's merge option to combine the changes in these files so no work is lost. For details, see "Merging Files," later in this chapter.

Using File Properties

Each file has several categories of properties associated with it.



To view file properties:

1. Select a file from the files list or the project tree.
2. Select **Properties...** from the Files menu or the context menu.

   Or,

   Press Alt+Enter.

   The File Properties dialog appears.

Working File Properties

The Working File page displays information about the working file. It tells you:

- The name and path to the working file.
- Whether the file currently resides in the working directory, and, if so, its size, time stamp, and status. For example, if the working directory does not contain the file, the file's status is Needs Check-out.
- The description of the file.

Vault File Properties

The Vault File page displays:

- The name of the vault file (with the .VVF extension).
- Its size in bytes.
- Whether it is compressed.
- Its storage type.
- The most recent version number.
- Whether the file is locked and who locked it.
- The number of non-permanent versions that can be stored in the file without being automatically deleted.

Other Properties

The History and Audit Trail pages display the history of the file and the entries for it from the audit log. These can also be viewed by selecting the file from the project tree then clicking the History or Audit tabs below the right pane of the project window. Each of them has a context menu from which you can perform operations. For example, you can compare versions of text files and review the properties for an audit entry.

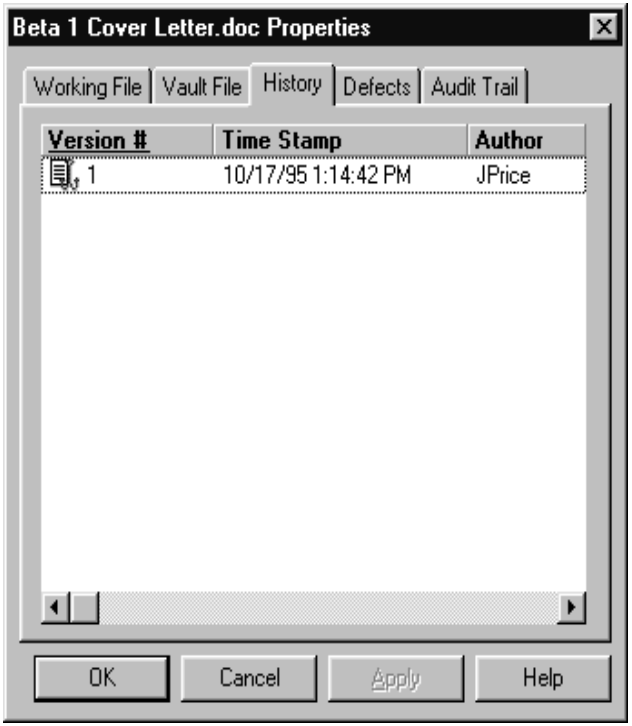
See the next section for more information about the History list.

See Chapter 6, “Using the Audit Log,” for more information about the Audit log.

Using the History List

When you select a file from the project tree, the Files tab changes to the History tab. Selecting the History tab displays the selected file’s version history in the right pane, and the History menu appears on the menu bar. However, the Filter list box and the scope buttons are disabled.

You may prefer to select the History tab from the File Properties dialog shown below. **(**Right-click to see a context menu.)



Icons

One icon precedes the name of each version. It indicates whether the version is:

 A non-permanent version that can be automatically deleted when the maximum number of non-permanent versions is exceeded.

 A permanent version that can never be automatically deleted.

Only non-permanent versions can be purged from a file, and at least one version must remain in a file after the purge—even if it is non-permanent.

Columns

The history list contains several columns, each with its own column header. The columns for the history list are:

Version #	Number Versions gave the version as it was checked in.
Time Stamp	Date and time of the version.
Author	Person who checked in the version.
File Size	Size of the version in bytes.
Reason for Check-In	Reason provided for checking in the version.

Sorting

Click a column header to sort the displayed versions based on the value in that column. Version # sorts from highest to lowest number. Time Stamp sorts from most recent to earliest time. Otherwise, the sort is in ascending order, numeric or alphanumeric depending on the data. To change the sort order from ascending to descending (or vice versa), click the header a second time. Right-click another header to perform a secondary sort.

Group bands

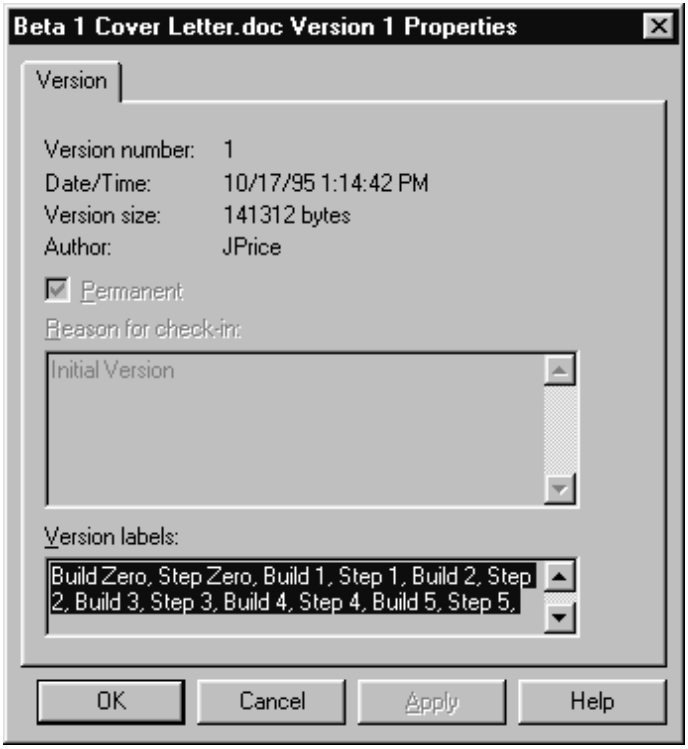
When sorting, group bands separate entries with the same value in the sort column. When the values of the sorted column are likely to be unique (such as a time stamp), no group bands appear.

Menus

After selecting one or more versions from the history list, use the History menu or right-click to display a context menu.

Double-click

Double-click a version to see its properties.



Version Properties

The properties for a Version include the same information that the icons and columns show. In addition, the Version property page shows all the version labels that have been associated with this version.

Adding Files to a Project

You can add files from any working directory on your workstation to the subproject that corresponds to that directory. The contents of the working files become the first versions of those files stored in the project vault. In fact, the reason for check-in for an added file is automatically “initial version.”

If the working directory for the subproject is not already on your workstation:

1. Select **Project** ⇨ **Subproject** ⇨ **Create Directory Structure...** from the menu bar.

   Versions automatically creates the path to the working directory for the subproject you selected. It also creates the working directories for the subprojects of your selection.

2. Move the files to be added to the project to the working directory or create them there.

TIP
If the administrator allows you to select your own working directories, you can specify more convenient locations on your workstation’s hard drive for the project or a subproject’s files. For example, you might want to use a shorter path or a different drive letter. To specify a working directory, select a subproject from the project tree. Then select **Properties...** from the Project menu or a context menu or press Alt+Enter.

To select the files to be added:

1. Select a subproject from the project tree.

2. Select the Files tab below the right pane.

   The files list appears in the right pane.

3. Select Files Not in Project from the Filter drop-down list box above the files list.

   The files list contains the names of any files in your working directory that are not already under version control.

4. Select the files to be added from the files list.

TIP

Use the All Descendants  or Immediate Descendants  scope buttons to select files across multiple subprojects. Use **Select All** on the Files or context menu to select all the files simultaneously.

To add files quickly using the current project and workstation defaults:

1. Select the files to be added.
2. Click Add Files  .

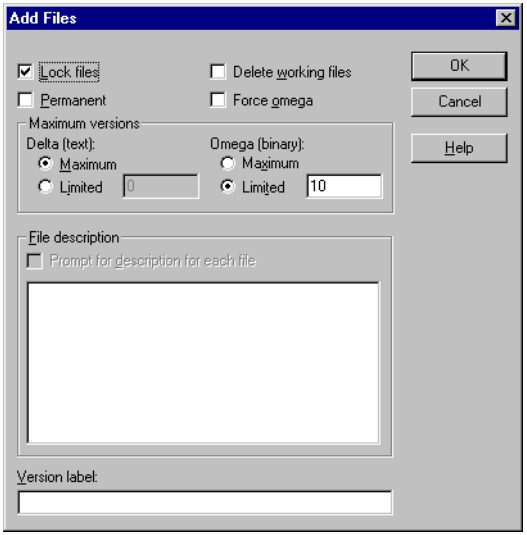
   Or,

   Select Add Files from the files list context menu.

To add the files and set options for them:

1. Select the files to be added.
2. Select **Files** ⇨ **Add Files...** from the menu bar.

   The Add Files dialog opens.



3. Select the Lock Files check box to lock the files (with you as the locker) or clear the box to leave the files unlocked at this time.

   When you lock a file, only you can work on it unless another team member breaks your lock. When project security is enabled, only an administrator can break your lock.

4. Select the Permanent check box to make the first versions of these files permanent or clear the box to make the versions non-permanent.

5. Select the Delete Working Files check box to delete the files from your workstation, storing them only in the project vault, or clear the box to leave the working files in the working directory.

6. By default, text files use delta storage, and binary files use omega storage. If the selected files are text files that are large and change frequently, select the Force Omega check box to apply omega storage to these files.

7. The number of non-permanent versions stored for any file in the project depends on the file's storage type. By default, Versions stores a maximum of 32767 versions of files using delta storage and ten versions for files using omega storage.

   To change the number of non-permanent versions stored:

   a. Select the Maximum or Limited option button for one of the storage types.

   b. If you selected Limited, type a maximum number of versions in the Limited text box.

   c. Repeat steps a and b for the other type of storage, if applicable.

8. Type a generic description for all the files in the File Description text box.

   Or,

   Select the Prompt for Description for Each File check box.

9. If retrieving these files later as a group or if retrieving this specific version of the one file being checked in will be important, type a label in the Version Label text box. For example, you may label a file as "technical review."

10. Click OK.

    The names of the selected files disappear from the files list and appear on the project tree.

    The Enter Description dialog opens if you selected the Prompt for Description for Each File check box. Type descriptions as necessary.

11. Change the filter above the files list to Files in Project so you can see the names of your files in the files list again.

Setting Version Control Options

Workstation options control how Versions behaves on a specific workstation. Several of these options are for version control. You can override these options as you check a particular file in or out by selecting **Files ⇨ Check In...** or **Files ⇨ Check Out...** from the menu bar and using either the Check In or Check Out dialog.

To set version control options for your workstation:

1. Select **Options ⇨ Workstation...** from the menu bar.
   The Workstation Options dialog opens.

2. Select the Defaults tab.

3. To lock files as you check them out, select the Lock Files on Check-Out check box. Otherwise the file's current lock status is preserved.

4. To retain your lock on a file after a check in, select Keep File Locked on Check-in.

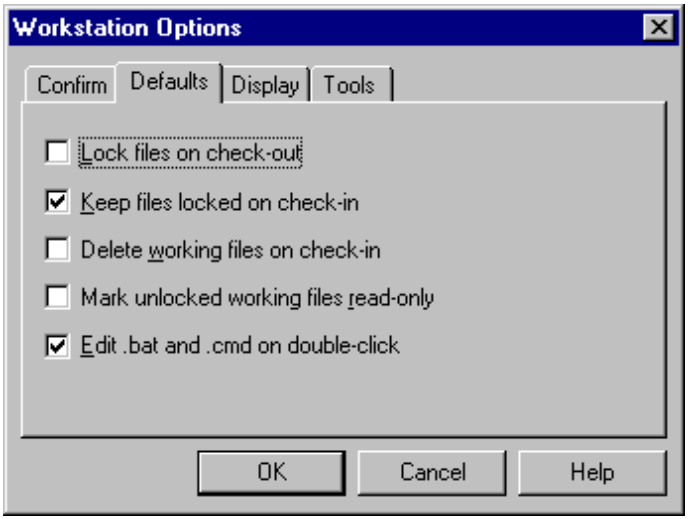
   Otherwise the working file is automatically unlocked.

5. To delete the working file as part of the check-in process, select Delete Working Files on Check-in.

   Otherwise the file is left in the working directory.

6. To protect yourself from accidentally updating files that you do not have locked, select Mark Unlocked Working Files Read-only.

   Otherwise, you may make changes to unlocked files. (Files accidentally changed in this way can be merged with the most recent checked-in version using **Files** ⇨ **Merge** from the menu bar.)



Locking and Unlocking Files

You must lock a file before you:

- Check it in.
- Change its maximum number of versions using **Files** ⇨ **Maximum Versions...**.
- Remove it from version control using **Files** ⇨ **Delete...**.
- Delete one of its versions using **History** ⇨ **Delete**.
- Purge non-permanent versions from it using **Files** ⇨ **Purge**.

- Change the status of one of its versions from permanent to non-permanent or vice versa using **History** ⇨ **Permanent** or using **History** ⇨ **Not Permanent**.

To lock a file:

1. Select one file from the project tree.

   Or,

   Select one or more files from the files list.

2. Select **Lock** from the Files menu or a context menu.

   Or,

   Click Lock  .

To unlock a file, use **Unlock** or click Unlock  .

Checking In Files

When a file is checked in, the contents of the working file are stored in the project vault as a new version of the file.

The first two procedures check files in quickly using:

- The current workstation and project version control options
- No reason for checking in the files

The last procedure displays the Check In dialog, shown below, so you can change those options or supply a reason for check-in.

To check in a single file quickly using the project tree:

1. Select a file from the project tree.
2. Select **Check In** from the context menu.

   The selected working file is added to the project vault as a new version.

TIP
Use the All Descendants  or Immediate Descendants  scope buttons to select files across multiple subprojects.

To check in files quickly using the files list:

1. Select one or more files from the files list.
2. Click Check In  .

   Or,

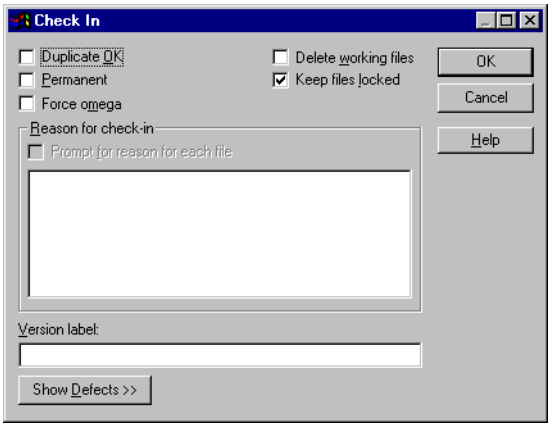
   Select **Check In** from the context menu.

   The selected working files are added to the project vault.

To check in files and set options for them:

1. Select one or more files from the files list.
2. Select **Files⇨Check In...** from the menu bar.

The Check In dialog opens.



3. Select the Duplicate OK check box if it is OK to create a version that is identical to the previous one. Clear the box to reject a duplicate version.

4. Select the Permanent check box to make the versions of these files permanent or clear the box to make the versions non-permanent.

5. By default, text files use delta storage, and binary files use omega storage.

   If the selected files are text files that are large and change frequently, select the Force Omega check box to apply omega storage to these files.

6. Select the Delete Working Files check box to delete the files from your workstation, storing them only in the project vault or clear the box leaving the working files in the working directory.

7. Select the Keep Files Locked check box to keep your lock on the checked-in files. Clear the box to unlock the files as part of the check-in process.

8. Type a generic reason for all the files in the Reason for Check-in text box using a maximum of 255 characters.

   Or,

   Select the Prompt for Reason for Each File check box.

9. If retrieving these files later as a group or if retrieving this specific version of the one file being checked in will be important, type a label in the Version Label text box. For example, you may label a file as “technical review.”

10. If you are using StarTeam Workstation with Versions, click Show Defects to view and edit files which contain defects.

11. Click OK.

    The Enter Reason dialog opens if you selected the Prompt for Reason for Each File check box. Type reasons as necessary.

Checking Out the Most Recent Version

When you check out a file using one of the following procedures, Versions copies the latest version of that file to the working directory for the subproject to which the file belongs.

If a copy of that file is already in the working directory, it is overwritten unless that copy appears to be more recent than the checked in version. Then you are asked to confirm the check out.

For example, if you check a file out, edit it, decide you don’t want your edits, and check out the file again, you are asked to confirm the check out.

The first two procedures check files out quickly using the current workstation options for check-out. The last procedure displays the Check Out dialog so you can change the Lock Status option or force a check-out when the latest version of a file is older than the working file.

To check out a single file quickly using the project tree:

1. Select a file from the project tree.

2. Select **Check Out** from the context menu.
   The latest version of the selected file is checked out.

TIP
Use the All Descendants or Immediate Descendants scope buttons to select files across multiple subprojects.

To check out files quickly using the files list:

1. Select one or more files from the files list.
2. Click Check Out .

   Or,

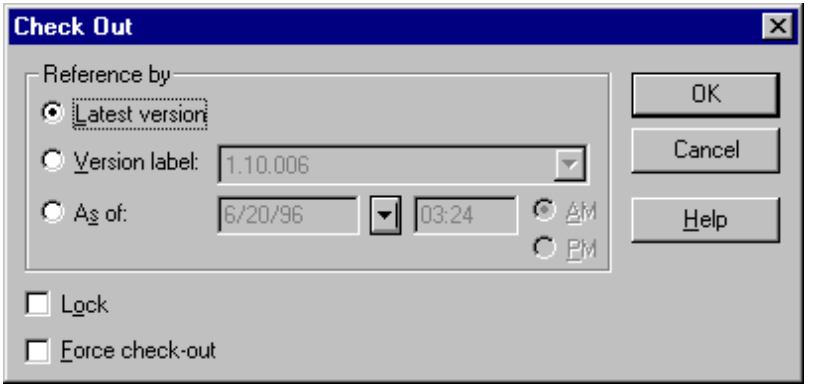
   Select **Check Out** from the context menu.

   The most recent versions of the selected files are checked out.

To check out files and set check-out options:

1. Select one or more files from the files list.
2. Select **Files** ⇨ **Check Out...** from the menu bar.

   The Check Out dialog opens.



3. Select the Latest Version option button.

4. Select the Lock check box if you want to lock the files. Clear it to retain each file's current lock status.

5. You can select Force Checkout to overwrite any working files, even if they are more recent than those being checked out.

Checking Out a Previous Version

When you check out a file using one of the following procedures, Versions copies the selected version of that file to the working directory for the subproject to which the file belongs.

The first procedure allows you to compare versions before you select the one you want. However, you must use the current workstation options for check-out.

The second displays the Check Out dialog so you can change the Lock Status option or force a check-out when the latest version of a file is older than the working file. It allows you to check out versions of more than one file.

To check out any previous version of a single file:

1. Display the history list by doing one of the following:
   - Right-click the name of the file from the files list.
     Select **Properties...** from the context menu.
     Then select the History tab from the File Properties dialog.
   - Select the name of the file from the project tree.
     Then select the History tab below the project window's right pane.

2. To compare two versions to determine which of the two to check out:

   a. Select two previous versions from the history list.

   b. Select **Compare** from the History menu or context menu.

3. Select the version to be checked out.
4. Click Check Out  .

   Or,

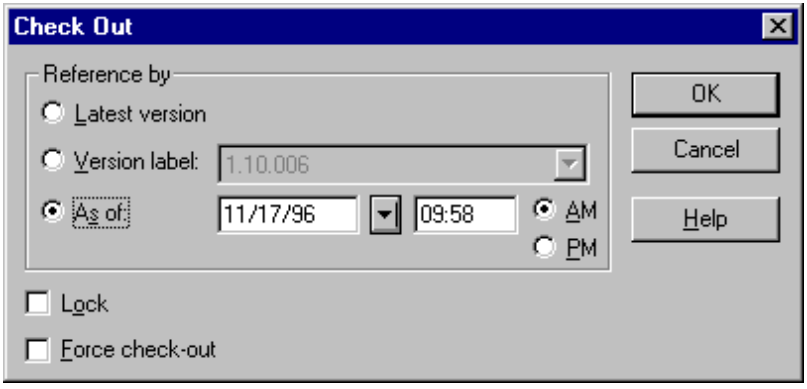
   Select **Check Out** from the History menu or context menu.

   The selected version is copied to the file's working directory using the current workstation options for check-out.

To check out previous versions of more than one file:

1. Select one or more files from the files list.
2. Select **Files** ⇨ **Check Out...** from the menu bar.

   The Check Out dialog opens. This allows you to check out the most recent version of the selected files. Or you can check out versions of those files with the same label or check-in date.



3. Indicate what versions you want by selecting one of the following option buttons:
   - Version Label to check out the versions with a specific label, which you select from the Version Label drop-down list box.
   - As Of to check out the versions checked-in just before the date and time you specify.

a. To specify the date, type the date in the text box.

   Or,

   Click the button between the date and the time to use the calendar. Specify the date using the next and previous month buttons and the arrow keys.

b. To specify the time, type the time in the text box, then select the AM or PM option button.

4. Select Lock to lock each file as part of the check-out process. Clear it to preserve the current lock status of each file.

5. Select Force Checkout to overwrite working files, even if they are more recent.

Using Keywords in Files

If Keyword Expansion is enabled for a project, you can embed keywords within a project text file. These keywords are automatically expanded during file check-out to provide file and version information within the file.

The keywords available in Versions are:

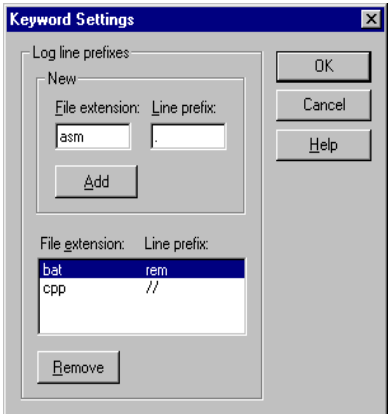
$Archive$	Fully qualified vault file name
$Author$	User who checked in the version
$Date$	Date and time stamp for the version
$Header$	Combination of Archive, Revision, Date, and Author
$Locker$	User who has the file locked (if any)
Log	File change history (see below)
$NoKeywords$	Turn off keyword expansion for the rest of the file
$Project$	Name of the project
$Revision$	Version number

$Subproject$	Name of the subproject
$Workfile$	Unqualified name of the working file (for example, foo.cpp)

Log Keyword Settings

Log is a special keyword because it expands to a multiline entry. The Log keyword expands to include information for each version of the file. Version history includes the Version Number, Date, Author, and Reason for Check In.

Using the Log keyword also allows you to place a line prefix or comment characters by file type in all expanded lines.To ensure that these lines are recognized as comments, you can attach a prefix to each line. You can provide different prefixes for files with different extension. For example, you can use “rem” for .bat files.



NOTE

You must have administrative privileges to assign file extension and line prefix pairs.

To add a file extension/prefix pair:

1. Select the root subproject from the project tree.
2. Select Properties... from the Project menu

   Or,

   Press Alt + Enter.

   The Project Properties dialog opens.
3. Select the Defaults tab.

   The Defaults property page opens.
4. Select the Keyword Expansion check box (if it is not already checked).
5. Click Settings...

   The Keyword Settings dialog opens.
6. Type the extension in the File Extension text box.
7. Type a prefix that will appear in each line of the Log entry in a file with the corresponding extension in the Line prefix text box.

NOTE

A file extension or line prefix can contain a maximum of 32 characters.

8. Click Add

   Your extension/prefix combination appears in the File extension/Line prefix list box.

To remove a file extension/prefix pair:

1. Select the File Extension and Line Prefix pair you want to delete from the list box.
2. Select Remove to delete this entry.

Checking Out a Build or Milestone

Both builds and milestones are sets of file versions with the same label. The milestone represents a significant step in the life cycle of the product and the versions are permanent. Normally, testers and other team members install a build or milestone to review it. However, sometimes you need the file versions used to create the build or milestone. For example, if a customer needs a fix for a particular release of software product, a developer might need to look at the source code that produced that release.

Before checking out a build or a milestone, make sure you check in any files you have been working on so the build or milestone does not overwrite those files.

To check out a build or a milestone:

1. Select the appropriate subproject from the project tree.

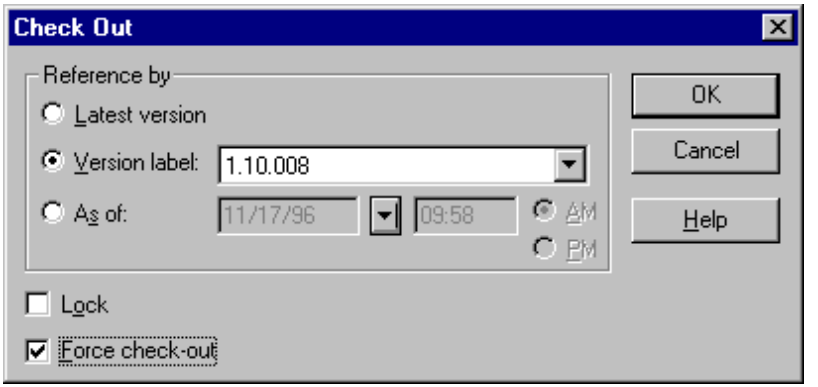
2. Select the Files tab below the right pane to list all the files.
   The files list appears in the right pane.

3. Use the Files in Project filter and the All Descendants  scope button, if appropriate.

4. Select **Files** ⇨ **Select All** from the menu bar.

5. Select **Files** ⇨ **Check Out...** from the menu bar.

The Check Out dialog opens.



6. Select version label.

7. Select the build or milestone's label from the Version Label drop-down list box.

8. Select the Force Check-out check box in case you have more recent versions of the selected files in your working directories.

   Only the versions of files that match the label are checked out.

Comparing Files

You can make any of the following comparisons between text files and/or text versions.

To compare two versions of a file:

1. Display the history list by doing one of the following:

   - Right-click the name of the file from the files list.
     Select **Properties...** from the context menu.
     Then select the History tab from the File Properties dialog.
   - Select the name of the file from the project tree.
     Then select the History tab below the project window's right pane.

2. Select two versions of a text file from the history list.
3. Click Compare  .

   Or,

   Select **Compare** from the History menu or context menu.

   Visual Diff, Versions's differencing program, displays the two versions, indicating the differences with different colors.
4. Select **File** ⇨ **Exit** from the Visual Diff menu bar to return to Versions.

To compare a version to the working file:

1. Display the history list as explained in step 1 above.
2. Select one version of a text file from the history list.
3. Click Compare  .

   Or,

   Select **Compare** from the History menu or context menu.

   Visual Diff displays the two files.
4. Select **File** ⇨ **Exit** from the Visual Diff menu bar to return to Versions.

To compare two working files that are text files:

1. Select a subproject from the project tree.
2. Select the Files tab below the right pane.

   The files list appears in the right pane.
3. Select two text files from the files list.

4. Click Compare  .

   Or,

   Select **Compare** from the Files menu or context menu.

   Visual Diff displays the two files.

5. Select **File** ⇨ **Exit** from the Visual Diff menu bar to return to Versions.

To compare a text file in the working directory with its most recently checked-in version:

1. Select a file from the files list or project tree.
2. Click Compare  .

   Or,

   Select **Compare** from the Files menu or context menu.

   Visual Diff displays the two files.

3. Select **File** ⇨ **Exit** from the Visual Diff menu bar to return to Versions.

See Chapter 12, “Using Visual Diff,” for information about searching for differences and other features of the Visual Diff application.

Merging Files

Although Versions provides file locking and communication tools to prevent team members from working on files simultaneously, it is not always possible to prevent these working collisions. One team member may begin to make changes before locking a file, only to realize too late that another team member has locked it and made changes as well. Or a team member working off-site might update an out-of-date file without realizing that there is a newer version stored in the project vault.

Versions provides a merge option to combine the changes in these files so that no work is lost. It merges an outdated working file with the most recently checked in version by comparing the two "new" versions with the version on which the outdated file was based (i.e., the version that was checked out to create the working file).

To merge an outdated working file with the most recently checked in version:

1. Select the file to be merged from the files list.

2. Lock the file by doing one of the following:

   - Click Lock  .
   - Select **Lock** from the Files menu or a context menu.

3. Select **Files** ⇨ **Merge** from the menu bar.

   Versions warns you when the files contain differences (conflicts) and notifies you when the files are the same.

   After the merge, a merged file, with the differences between the two files clearly marked, becomes your working file.

4. Edit the merged file to review the changes and resolve conflicts before checking the file in.

Sample of conflict output in merged file:

```
>>>> === An editing conflict was detected in lines 10 through 10 ===
>>>> The original (root) file contains the following:
BUFFERS=10,0
>>>> The last checked-in version contains the following:
BUFFERS=20,0
>>>> The current working file contains the following:
BUFFERS=30,0
>>>> === End of the conflict section ===
```

You can use stcmd mrg at the command-line (and not **Files** ⇨ **Merge**) to merge branched subprojects. See Chapter 13, "Using the Command-line Interface," for more information.

Compressing and Decompressing Vault Files

Versions allows you to compress or decompress one or more vault files. Compressing vault files:

- Reduces disk storage requirements for those files
- Can improve check-in and check-out performance because it reduces disk input and output

To compress or decompress vault files:

1. Select one or more files to be compressed from the files list.
2. Select **Files** ⇨ **Compress Vault File** or **Files** ⇨ **Decompress Vault File** from the menu bar.

Modifying File and Version Properties

You can modify the following file and version properties if you have the file locked or checked out. If security is enabled, only an administrator can change some of the properties.

The file properties you can modify are:

- Description
- Maximum number of versions (administrator only)
- File Name

The version properties you can modify are:

- Status (Anyone can change the status from non-permanent to permanent, but only an administrator can change it from permanent to non-permanent.)
- Reason for check-in

You can also review a file's history and audit log from the File Properties dialog.

To modify file properties:

1. Select a file from the project tree. Then select **Properties...** from the Project menu or the context menu.

   Or,

   Select a file from the files list. Then select **Properties...** from the Files menu or the context menu.

   The File Properties dialog opens.

2. Select the appropriate tab: Working File or Vault File, History, or Audit Trail.

3. Change the setting for the property.

To modify version properties:

1. Display the history list by doing one of the following:

   - Right-click the name of the file from the files list.
     Select **Properties...** from the context menu.
     Then select the History tab from the File Properties dialog.
   - Select the name of the file from the project tree.
     Then select the History tab below the project window's right pane.

2. Double-click the version in the history list.

   Or,

   Select a version, then select **Properties...** from the History menu or context menu.

   The Version Properties dialog opens.

3. Change the setting for the property.

Deleting a Version

When project security is enabled, only an administrator can delete a version of a file from the project vault. Use this procedure with caution. The history information for all deleted versions remains in the project database file, even though the deleted versions are not recoverable. In the file's history display, the history information for deleted versions is dimmed.

To delete versions:

1. Select the file from the project tree.

2. Select the History tab below the right pane.

   The history list appears in the right pane.

3. Select one or more versions to be deleted.

4. Select **History**⇨**Delete** from the menu bar.

   Or,

   Press Delete.

   For permanent versions, Versions asks for confirmation.

Deleting a File

When project security is enabled, only an administrator can remove a file from version control. Even when security is disabled, you must lock the file before you can remove it. Removing a file destroys all the information about its version history and audit entries. Use this procedure with caution.

To remove one or more files from version control:

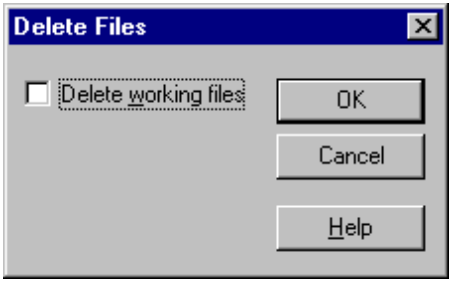
1. Select a subproject from the project tree.
2. Select the Files tab below the right pane.

   The files list appears in the right pane.
3. Select the files to be removed from version control from the files list.
4. Select **Files⇨Delete...** from the menu bar.

   Or,

   Press Delete.

   The Delete Files dialog opens.



5. Select the Delete Working Files check box to delete the corresponding working files from their working directories. Clear the box to leave the working files undisturbed.

Sending Versions

You can attach copies of versions to e-mail messages using **Files** ⇨ **Send To...** and **History** ⇨ **Send To...**. See Chapter 7, “Using E-mail with Versions” for more information.

Reporting and Charting

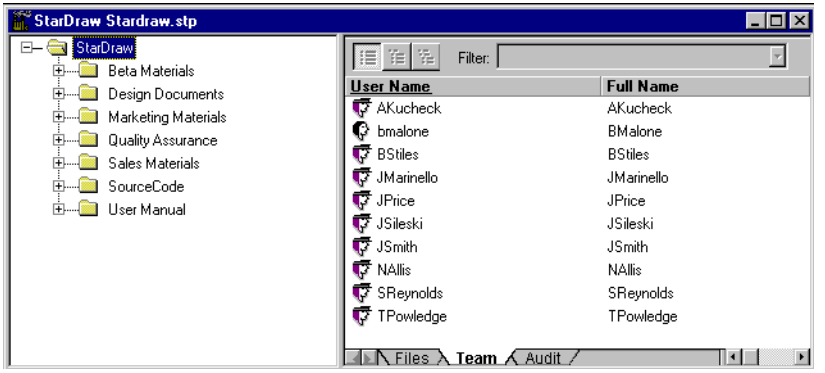
You can create a number of charts and reports using **Files** ⇨ **Reports...** and **Files** ⇨ **Charts...**. See Chapter 8, “Reporting,” and Chapter 9, “Charting,” for more information.

Chapter 5
Managing the Team

If you select the Team tab, the team list displays in the right pane. The Team menu also becomes available from the menu bar.

The team list displays each user who is a member of the project team. It contains several columns, each with its own column header.

The Filter drop-down list box and the scope buttons are disabled.



TIP
Right-click to see context menus.

Icons

One icon precedes the name of each user. It indicates the user's status:

 Administrator

 Non-administrator

One icon precedes the name of each group.

 Group

Columns

The columns for the team list are:

User Name	The name used to log on to the project.
Full Name	The name used for e-mail.
Admin	Indicates (with a Yes or No) whether the user is an administrator.
User Type	Indicates (with User or Group) whether the user entry is an individual or a group users.

Sorting

Click a column header to sort the displayed users based on the value in that column. Admin sorts Yes to No. User Type sorts User to Group. User Type sorts User to Group. Otherwise, the sort is in ascending, alphanumeric order. To change the sort order from ascending to descending (or vice versa), click the header a second time. Right-click another header to perform a secondary sort. The primary sort column's header is underlined.

When sorting, group-bands separate entries with the same value in the sort column. When the values of the sorted column are likely to be unique, no group bands appear.

Menus

After selecting one or more users, use the Team menu or right-click to display a context menu.

Double-click

Double-click an entry to see its properties.

Adding a User

Any number of users can access a project, but only after they have been added to the list of team members.

When project security is enabled, only an administrator can add users. Otherwise users can add themselves as they log on to the project.

If a user is added when project security is disabled, that user has a blank password. If a project's security is enabled later, that user continues to have a blank password until an administrator or the user changes it.

As you add a user to the project (and if the project has e-mail enabled), VersionsVersions automatically sends e-mail notification to that user about being added to the project team. The e-mail is sent using the user's Full Name as the address.

To add a user:

1. Select the Team tab below the right pane.

   The team list appears.

2. Select **Team** ⇨ **New User...** from the menu bar.

   The New User dialog opens.

3. Type the name that will allow the user to log on to the project in the User Name text box. This is usually the user's logon name for the network. Select this name carefully, because it cannot be changed.

4. Type the user's e-mail name in the Full Name text box to use MAPI-compliant e-mail from Versions. For example, when a new user is added to a project that has mail enabled, Versions sends a welcome mail message to that user.

   When using Microsoft Exchange, the full name and the Exchange profile name should be the same. Otherwise Microsoft Exchange returns an error message each time you log on to the Versions project. If this occurs, create a new Exchange profile with the same name as your Versions full name.

5. If project security is enabled, type the password that will allow the user to access the project in the Password text box.

6. If you typed a password, retype it in the Confirm text box to prove it was typed correctly at step 5.

7. Select the Administrator check box if this user will have administrator privileges.

8. Select the Force Change Password check box if you want the new user to change the password when logging on for the first time.

Adding a Group

A group consists of any number and combination of team members. You use groups to assign privileges to many users at one time instead of each user individually. When project security is enabled, only an administrator can create or modify groups.

Suppose you create a group called Testers that contains all the team members who will be testing the product. Suppose you also have a security template that allows team members to delete files. By assigning the Testers group to that security template for the Automated Testing subproject, you give each tester the ability to delete files in that subproject. As testers come and go, you add or delete their names from the Testers group.

See the “Setting Up Subproject Level Security” section in Chapter 3 for more information about security templates, assigning security templates to subprojects, and assigning users and groups to security templates.

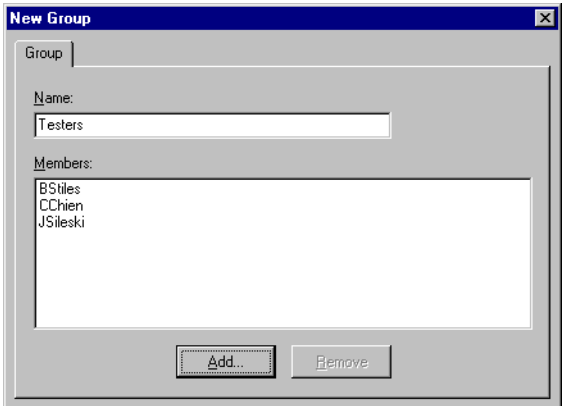
To create a group:

1. Select the Team tab below the right pane.

   The team list appears.

2. Select **Team** ⇨ **New Group...** from the menu bar.

   The New Group dialog opens.



3. Type the name for the group in the Name text box.

4. Click Add....

   The Add Users dialog opens.

5. Select one or more users then click OK.

Controlling Group Membership

You can add users to and remove them from groups using either the Group Properties dialog or the Group tab of the User Properties dialog.

Use the Group Properties dialog to make one or more user changes to a group. For example, several users might need to be added to an existing group at one time.

Use the Group tab of the User Properties dialog to modiyf the list of groups to which a user belongs. For example, a new team member might need to be added to several groups at once.

To change an existing group's membership list:

1. Select the Team tab below the right pane.

   The team list appears.

2. Select the group to be modified.

3. Select **Properties...** from the Team or context menu.

   The Group Properties dialog opens.

4. To add users: Click Add... to select additional members from the Add Users dialog (which lists all the team members who are not in this group).

   Or,

   To remove users: From the Members list box, select one or more users to be deleted, then click Remove.

To change the list of groups to which a user belongs:

1. Select the Team tab below the right pane.

   The team list appears.

2. Select the user to be modified.

3. Select **Properties...** from the Team or context menu.

   The User Properties dialog opens.

4. Select the Group tab.

   The Group property page opens.

5. To add groups: Click Add... to select additional groups from the Add Groups dialog (which lists all the groups that do not include this user yet).

   Or,

   To remove groups: From the Membership list box, select one or more groups to which the user should not belong. Then click Remove.

Changing Your Password

When project security is enabled, you can change your personal password.

To change your password:

1. Select the Team tab below the right pane.

   The team list appears in the right pane.

2. Double-click your name in the team list.

   Or,

   Select your name. Then select **Properties...** from the Team menu or the context menu.

   The User Properties dialog opens.

3. Type the new password in the Password text box.

   Passwords are case-sensitive and must contain at least six characters.

4. Confirm the new password in the Confirm text box.

Modifying User Properties

If you are an administrator, you can modify properties for any user. You can:

- Change a user's full name, which is used for e-mail purposes.
- Change a user's password.
- Change a user's status from non-administrator to administrator or vice versa.
- Force a password change during the user's next log on attempt.

To modify user properties:

1. Select the Team tab below the right pane.

   The team list appears.

2. Double-click the user.

   Or,

   Select the user, then select **Properties...** from the Team menu or context menu.

   The User Properties dialog opens.

3. Change the setting for the property.

Deleting Users or Groups

When project security is enabled, only an administrator can delete users or groups.

To delete users or groups:

1. Select the Team tab below the right pane.

   The team list appears.

2. Select the users and/or groups to be deleted.

3. Select **Team** ⇨ **Delete** from the menu bar.

   Or,

   Press Delete.

Reporting and Charting

You can create charts and reports using **Team** ⇨ **Reports...** and **Team** ⇨ **Charts...**. See Chapter 8, “Reporting,” and Chapter 9, “Charting,” for more information.

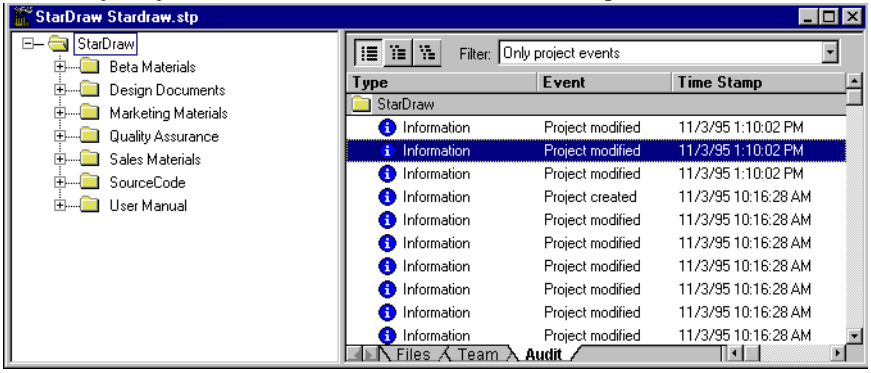
Chapter 6
Using the Audit Log

The audit log is a chronological record kept by Versions. It accumulates data about actions performed on projects, subprojects, files, and users. It appears when you select the Audit tab below the right pane.

It lists the entries that are all of the following:

- Attached to the selection from the project tree
- Match the filter selected from the Filter drop-down list box
- Match the depth specified by the selected scope button

The following figure shows all the audit entries attached to the root project, matching the filter Only Project Events, and at the No Descendants depth.



Audit Trail tabs

Most property dialogs have an Audit Trail tab that allows you to view the log for the selected object. For example, you can view the entries in the audit log that pertain to a specific version—a subset of the entries that are normally displayed in the log for a project, subproject, or file.

Columns

The Audit Log contains several columns, each with its own column header. The columns are:

Type — The type of action:

Information
Information audit entries are generated for each normal project activity, such as adding a file to a project and checking in a new version.

Warning
Warning audit entries are generated for non-critical project activity. For example, attempting to check in the file generates a warning when your software "touches" a file to change the date and time stamp, but the contents of the file are unchanged.

Error
Error audit entries are generated for critical abnormal project activity.

Event — Name of the action, such as File locked, User added, Project modified, or Subproject deleted. See "Audit Log Events," later in this chapter for more details.

Time Stamp — Date and time of the action.

User — Person performing the action.

Description	Description of the action.
Owner	Project, subproject, file, version or user receiving the action.

Sorting

Click a column header to sort the displayed entries based on the value in that column. Time Stamp sorts from most recent to earliest in time. Otherwise, the sort is in ascending order, numeric or alphanumeric depending on the data. To change the sort order from ascending to descending (or vice versa), click the header a second time. Right-click another header to perform a secondary sort. The primary sort column's header is underlined.

Group bands

When sorting, group bands separate entries with the same value in the sort column. When the values of the sorted column are likely to be unique, no group bands appear.

Menus

After selecting one or more entries, use the Audit menu or right-click to display a context menu.

Double-click

Double-click an entry to see its properties. This shows you the contents of the row in a dialog.

Using Audit Log Filters

Filtering allows you to limit the kinds and quantity of entries to be displayed in the audit log. Select a filter from the Filter drop-down list box above the audit log. What is displayed is also affected by your selection from the project tree and the scope buttons. The filters are:

<Show All>	Displays all the entries.
Date Range = 1 Month	Displays only the entries that were created within the last month.
Date Range = 1 Week	Displays only the entries that were created within the last week.
Date Range = Last 24 hours	Displays only the entries that were created within the last 24 hours.
Only file events	Displays only the entries about actions that were performed on files.
Only project events	Displays only the entries about actions that were performed on projects
Only subproject events	Displays only the entries about actions that were performed on subprojects.
Only user events	Displays only the entries about actions that were performed on the user list.
Type = Error	Displays only the entries that record errors.
Type = Information	Displays only the entries that record information.
Type = Warning	Displays only the entries that record warnings.
User = Me	Displays only the entries about actions performed by the logged on user.

Audit Log Events

Events are actions performed on an owner. For example, a file can be checked in or removed from version control. Such events are recorded in the audit log.

Audit events that can be attached to files are:

- File added
- File checked in
- File deleted (working file)
- File locked
- File modified
- File moved
- File purged
- File removed (from project)
- File unlocked

Audit events that can be attached to projects are:

- Build designated
- Label created
- Label deleted
- Milestone designated
- Project created
- Project modified
- Subproject added
- Subproject branched
- Subproject deleted
- Subproject modified
- Subproject moved
- User added
- User deleted

- User modified
- Group added
- Group deleted
- Group modified
- Security template added
- Security template deleted
- Version modified
- Version deleted

Enabling and Disabling the Audit Log

When security is enabled, only an administrator can enable or disable the collection of audit information.

To enable or disable auditing:

1. Select the root project from the project tree.
2. Select **Properties...** from the Project menu or the context menu.

   The Project Properties dialog opens.
3. Select the Audit Log check box to continue or clear it to stop adding entries to the audit log.

Clearing the Audit Log

When project security is enabled, only an administrator can clear the audit log. You may occasionally want to empty all or part of it.

To empty the entire audit log:

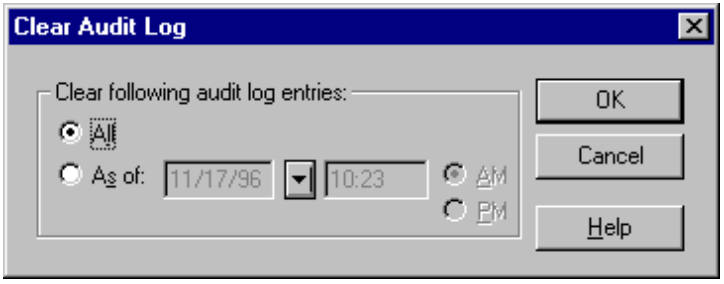
1. Select the root project from the project tree.

2. Select the Audit tab below the right pane.

   The Audit Log appears in the right pane.

3. Select **Audit** ⇨ **Clear Log...** from the menu bar.

   The Clear Audit Log dialog opens.



4. Do one of the following:

   - To clear the entire log, leave the All option button selected.
   - To delete everything before a certain date, select the As Of option button.

     a. To specify a date, use the calendar (available from the list box), the next and previous month buttons, and the arrow keys.

     b. To specify a time, type the time in the text box, then select the AM or PM option button.

Sending Audit Entry Descriptions

You can send the text of the descriptions for the selected audit entries as an e-mail message using **Audit** ⇨ **Send To...**. See Chapter 7, “Using E-mail with Versions” for more information.

Reporting and Charting

You can create a number of charts and reports using **Audit** ⇨ **Reports...** and **Audit** ⇨ **Charts...**. See Chapter 8, “Reporting,” and Chapter 9, “Charting,” for more information.

Chapter 7
Using E-mail

If you have a MAPI-compliant e-mail system, such as Microsoft Exchange, Versions uses your Versions full name to log on to e-mail and send mail messages.

You can:

- Send e-mail messages to other team members from within Versions
- Attach versions of files to e-mail messages
- Send the text of audit entries and reports

Versions automatically:

- Sends messages welcoming new users as they are added to Versions
- Notifies the former locker of a file when that file's lock has been broken

☆ **NOTE**
If a user has the same name in multiple address lists (for example, Personal and Global Address Lists), Versions uses the first one it finds based on the e-mail application's settings. For Microsoft Exchange users, address settings can be found in **Tools** ⇨ **Options...** ⇨ **Addressing**. The Addressing property page allows a user to define the order in which Exchange searches lists when multiple lists are present.

Enabling or Disabling E-mail

You can only use the mail features described in this chapter if e-mail is enabled for the project. When project security is enabled, only an administrator can enable or disable e-mail.

To enable or disable e-mail:

1. Select the project from the project tree.
2. Select **Properties...** from the Project menu or the context menu.

   The Project Properties dialog opens.
3. Select the Defaults tab.
4. Select the Use Mail check box to use your e-mail application from Versions or clear it to stop using e-mail.

If you enable mail, you must close and reopen the project for your change to take effect.

Changing a User's E-mail Name

The full name for a Versions user should be the name that the user uses for e-mail. You can always change your own full name, but when project security is enabled, only an administrator can change the full names of other users.

To change the user's e-mail name:

1. Select the Team tab below the right pane.

   The team list appears.

2. Double-click the user.

   Or,

   Select the user. Then select **Properties...** from the Team menu or the context menu.

   The User Properties dialog opens.

3. Type the new e-mail name in the Full Name text box, then click OK.

Sending Notes to Team Members

You can send a note to one or more team members using Versions.

To send a message:

1. Select **Project** ⇨ **Send Note To...** from the menu bar.

   The Send Note To dialog opens.

2. Select one or more recipients from the Recipient List.

3. Click Goto Mail....

   Continue writing the message using your MAPI-compliant e-mail application.

Sending File Versions

You can send any file version as an e-mail attachment. When you select files from the project tree or files list, you sent the most recently checked-in versions. When you select a version from the history list, you send that version, no matter how recent it is.

To send a version as an attachment:

1. Select the name of a file from the project tree.

   Or,

   Select one or more files from the files list.

   Or,

   Select a version from the history list.

2. Click Send  .

   Or,

   Select **Send To...** from the Files menu or a context menu.

   The Send To dialog opens.

3. Select one or more project users from the Recipients list. If you don't select any users, the mail message goes to all of them.

4. Click Send Now to send the message, which has the subject "Files from Versions Project: project name" or "Version from Versions Project: project name" and contains only the attached file or version.

   Or,

   Click Goto Mail... to edit the message inside your e-mail application. Then send the message via that application.

Sending Audit Entry Descriptions

You can send the text of the descriptions for the selected audit entries as an e-mail message.

To send audit entries via e-mail:

1. Select the name of a project, subproject, or file from the project tree.
2. Select the Audit tab below the right pane.

   The audit log appears in the right pane.
3. Select the audit entries to be sent from the audit log.
4. Click Send  .

   Or,

   Select **Send To...** from the Audit menu or context menu.

   The Send To dialog opens.
5. Select one or more project users from the Recipients list. If you don't select any users, the mail message goes to all of them.
6. Click Send Now to send the message, which has the subject "Audit entries from Versions Project: project name" and contains descriptions of the selected entries.

   Or,

   Click Goto Mail... to edit the message inside your e-mail application. Then send the message via that application.

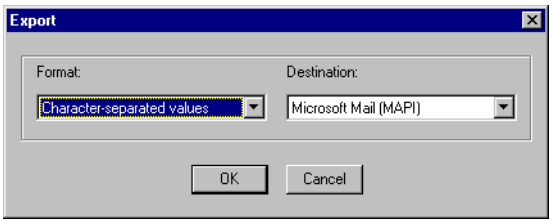
Sending a Report

You can send a report via e-mail as an attached file in a specified format, such as that of an Excel or Lotus 1-2-3 spreadsheet or a Word for Windows document.

To send a report via e-mail:

1. While previewing the report, click Send Report  at the bottom the Preview window to export the report.

   The Export dialog opens.



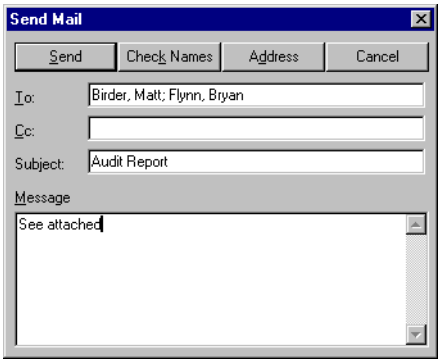
2. Select a format from the Format drop-down list box.

3. Make sure that Microsoft Mail (MAPI) appears in the Destination drop-down list box, then click OK.

4. Depending on the selected format, another dialog may open so you can select additional options, then click OK.

   For example, the Number and Date Format dialog opens when you select Data Interchange Format (DIF). On the other hand, if the selected format is Rich Text Format (RTF), no dialog opens.

Finally, the Send Mail dialog opens.



5. Provide addresses, a subject, and an explanatory message, then click Send.

   Versions sends the report in the selected format to the addressees.

   The report file is given a default name, such as filehist.chr or filesum.chr, depending on what report you created.

Chapter 8
Reporting

Versions offers a wide variety of reports for files, versions, users, and audit entries. Before you create a report, limit what is displayed in the right pane to fit that report's purpose. Then, if appropriate, select only the displayed items you want in the report.

Sorting, grouping, and selecting items in the right pane of the Versions project window affects report generation. To produce a report based on a specific sorting or grouping of Versions data, or to produce a report based on a limited or expanded set of objects, arrange the data in the Versions project window before you generate the report.

For example, a manager might want a report of all the files and their version histories for the entire project. She selects the root subproject from the project tree, the Files tab, the Files in Project filter, and the All Descendants scope button. She clicks the header of the Timestamp column to sort the files by date and time. Then she selects **Files ⇨ Reports...** and File Detail to create a File Detail Report.

A team leader might prefer to see the version histories for the files is a single subproject. He selects the subproject that corresponds to his team from the project tree and creates a File Detail Report much like the manager's.

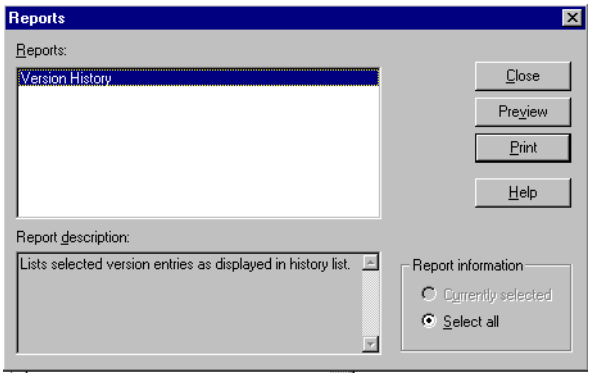
Individual programmers will want to see version histories only for the files they created. They perform the same process as the team leader, but they sort the files by Author and select only the files that have themselves listed as the Author. Then they use **Files ⇨ Reports...** to create a File Detail Report.

While these examples all pertain to files, you can also create useful reports about users and the audit log.

To create reports:

1. Select **Reports...** from the Files, History, Team, or Audit menus.

   The Reports dialog displays a list of possible reports in the Reports list box and a description of each report in the Report Description list box.



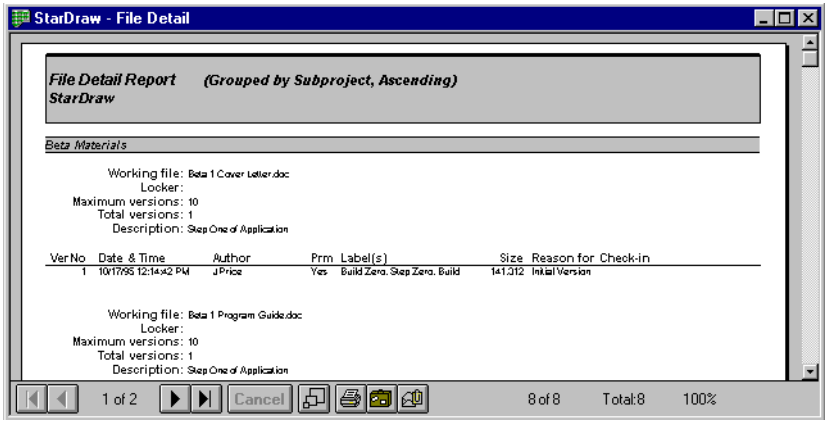
2. Select a report from the Reports list box.

3. Select the Currently Selected option button in the Report Information group box to create the report using only the items selected from the right pane. Select the Select All option button to use all the items displayed in the right pane.

4. Click Preview to review the report.

5. Click Print to print the report.

6. You can repeat steps 2 through 5 to create more reports. Click Close when you are done.



The report preview window's toolbar allows you to:

- Read status information about the number of pages and items included in the report. In the figure above, the items are defects. For example, the report in the figure above contains all 15 (or 100%) of the defects displayed in the defect list.

 Go to the first page

 Go to previous page

 Go to next page

 Go to the last page

 Print the report

 Write the report to a file

 Mail the report

Zoom in on the report
Switch among two or three of the following views:

- Normal width
  You may have to scroll horizontally and vertically to see rest of the page.
- Adjusted width (to fit the page's complete width in the window)
  The adjusted width view is not displayed if the window is large enough to show the page's complete width in normal width.
  You may have to scroll vertically to see the rest of page.
- Adjusted height (to fit the page's complete height in the window)

Available Reports

You can preview and print a number of different reports, depending on the selected tab.

Table 1: Available Reports

Tab	Name of Report	Description of Report
Files	File Detail	Lists files and their version histories.
	File Summary	Lists file summary information.
History	Version History	Lists selected version entries.
Team	User List	Lists selected user entries as displayed in team list
Audit	Audit List	Lists all audit entries as displayed in the audit log.

Writing a Report to a File

You can also write a report to a disk file in a specified format, such as that of an Excel or Lotus 1-2-3 spreadsheet or of a Word for Windows document.

To write a report to a file:

1. While previewing the report, click Write Report to File  at the bottom the Preview window to export the report.

   The Export dialog opens.

2. Select a format from the Format drop-down list box.

3. Make sure that Disk File appears in the Destination drop-down list box, then click OK.

4. Depending on the selected format, another dialog may open so you can select additional options, then click OK.

   For example, the Number and Date Format dialog opens when you select Data Interchange Format (DIF). On the other hand, if the selected format is Rich Text Format (RTF), no dialog opens.

   Finally, the Choose Export File dialog opens.

5. Provide a name and path for the disk file, then click Save.

   Versions writes the report to the selected file.

Sending a Report

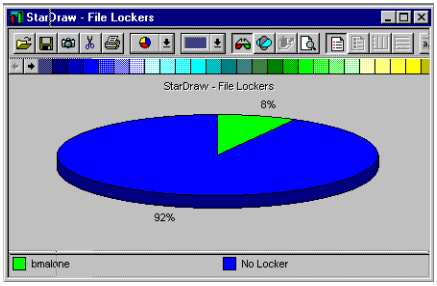
You can send a report via e-mail as an attached file in a specified format, such as that of an Excel or Lotus 1-2-3 spreadsheet or a Word for Windows document. You can send the report along with an explanatory note to any number of addressees. For more information about e-mail, see Chapter 7, “Using E-mail with Versions.”

Chapter 9
Charting

Versions offers a great variety of charts for files, versions, users, and audit entries. Before you create a chart, limit what is displayed in the right pane to fit that chart's purpose. Then, if appropriate, select only the displayed items that you want to chart.

Sorting, grouping, and selecting items in the right pane of the Versions project window affects chart generation. To produce a chart based on a specific sorting or grouping of Versions data, or to produce a chart based on a limited or expanded set of files, arrange the data in the Versions project window before you generate the chart.

For example, the administrator may want to make sure that the latest version of every file is checked in and unlocked before he executes a build. He selects the root subproject from the project tree, the Files tab, the Files in Project filter, and the All Descendants scope button. Then he selects **Files** ⇨ **Charts...** and creates a File Lockers chart. This chart shows who still has files locked.



To create charts:

1. Select **Charts...** from the Files, History, Team, or Audit menus.

   The Charts dialog displays a list of possible charts in the Charts list box and a description of each chart in the Chart Description list box.

2. Select a chart from the Charts list box.

3. If the Time Series drop-down list box is enabled (as it is for some charts), select Daily, Weekly, or Monthly as the time period for the charted information.

4. Select the Currently Selected option button in the Chart Information group box to create the chart using only the items selected from the right pane. Select the Select All option button to use all the items displayed in the right pane.

5. Click Chart to display the chart.

   Once the chart appears, you can use Chart help to change the chart in a variety of ways.

6. You can repeat steps 2 through 5 to create more charts. Click Close when you are done.

Available Charts

You can display a number of different charts, depending on the selected tab.

Table 2: Available Charts

Tab	Name of Chart	Description of Chart
Files	File Authors	Pie chart; number of files added to project by each user.
	File Lockers	Pie chart; number of files locked by each user.
	File Sizes	Double series bar chart; compares sizes of working files (first series) to sizes of project vault files (second series).
	File Versions	Bar chart; number of versions currently stored in the vault for each file.
	Files in Subprojects	Pie chart; number of files in each sub-project.
History	Version Authors	Pie chart; number of versions by each author.
	Version Sizes	Line chart; size of each version.
Team	Admin Users	Pie chart; numbers of administrators and non-administrators.
	User Types	Pie chart; numbers of users and groups.

Table 2: Available Charts

Tab	Name of Chart	Description of Chart
Audit	Audit Entry Events	Pie chart; number of entries for each event.
	Audit Entry Owners	Pie chart; number of entries for each owner.
	Audit Entry Types	Pie chart; number of entries for each type (Information, Warning, and Error).
	Audit Entry Users	Pie chart; number of entries for each user.

The Chart Window

The chart window displays the chart generated from your Versions data. Once you generate the chart, you can manipulate it in several ways, copy the chart or the data to the Windows Clipboard, and print the chart.

The chart window, by default, displays a toolbar and a pallet bar. It may, optionally, display a pattern bar.

The charting toolbar buttons are:

Import Chart
Open a chart file that has been previously saved to disk.

Export Chart
Save a chart file to disk.

Copy to Clipboard as Bitmap
Copy the chart to the Windows Clipboard as a bitmap.

Copy to Clipboard as Text
Copy the chart data to the Windows Clipboard as text.

Print Chart
Print the chart.

Change Gallery Type
Open a drop-down list box that allows you to change the type of chart (bar, scatter, pie, etc.). The button changes to indicate the current selection.

Change Color
Open a drop-down pallet that allows you to change the active color. The color can then be dragged to any chart element that accepts color.

Switch between 3D and 2D
Toggle between three-dimensional and two-dimensional charts.

Rotate Chart
Open a dialog that allows you to change the viewing angle and perspective of a three-dimensional chart.

Z-Clustered Series
Add depth to a multi-series chart.

Zoom
Enlarge a specific area of the chart.

Show or Hide Legend
Toggle the display of the chart legend on and off.

Show or Hide Series Legend
Toggle the display of the series legend on and off.

Vertical Grid
Toggle the display of a vertical grid on and off.

Horizontal Grid
Toggle the display of a horizontal grid on and off.

Edit Titles
Edit title text.

Change Text Fonts
Select a font for chart labels and legends.

Tools
Open a drop-down list box that allows you to toggle display elements such as the toolbar, pallet bar, and pattern bar on and off.

Change Chart Options
Open a multi-tab dialog in which you can set chart type and appearance, scale, view, and title text.

Editing Charts

After you generate a chart, you can change its appearance, change the chart data, copy the chart as a bitmap or as text, and print the chart.

Changing a Chart Type

Versions generates a chart type appropriate for the selected data. Available chart types include pie, bar, scatter, area, surface, and more. You can change the chart type to suit your needs.

To change the chart type:

1. Click Change Gallery Type  .

   A drop-down value set of possible types opens.

2. Select a chart type.

Changing Chart Data

Once you generate a chart you can change the data to perform “what if” calculations or to fine-tune the chart.

To change the chart data:

1. Click Tools  .

   The Tools drop-down menu opens.

2. Select Data Editor from the menu bar.

   The Data Editor window appears in place of the chart. The Data Editor lists each value used to create the chart.

3. Edit the values as necessary.

   a. Double-click a value to be changed.

      The value is highlighted and the alignment changes from right-aligned to left-aligned.

   b. Type the new value.

   c. Press tab to move through the data and make changes.

      Or,

      Double-click another value to be changed.

   d. Repeat step c as often as necessary until you are finished editing data.

4. Click Tools  .

   The Tools drop-down menu opens. There is a checkmark next to the Data Editor item.

5. Select Data Editor from the menu bar to remove the checkmark from the menu bar, close the Data Editor window, and regenerate the chart.

Changing Chart Color or Pattern

You can change the color of any chart element, including individual slices of a pie chart, bars on a bar chart, chart background, legend background, and so on by dragging color from either the Change Color toolbar button or the color pallet. You can also drag-and-drop patterns from the pattern bar onto any chart element that accepts color or patterns.

To change the color of a chart element:

1. Click Change Color  .

   A drop-down pallet opens.

2. Select a color from the pallet.

   The pallet closes and the color on the button changes to reflect your selection.

3. Drag the color from the button to the appropriate chart element.

To display the pallet bar and/or the pattern bar:

1. Click Tools  .

   A drop-down menu opens. Items on this menu are toggles. Selecting an item without a checkmark activates the option and places a checkmark next to the menu item. Selecting the item again deactivates the option and removes the checkmark from the menu bar item.

2. Select Pallet Bar or Pattern Bar to display the element.

   Once the pallet bar and/or pattern bar are displayed, you can drag-and-drop a color or pattern onto any chart element that accepts color or patterns (more chart elements accept color than patterns).

To remove a pattern from a chart element:

1. Click Change Chart Options  .

   The Chart Properties tab dialog opens with the General tab selected.

2. Click Color Scheme  .

A drop-down value set opens. Color scheme options are:

 Solid color

 Black and White pattern

 Color pattern

3. Select Solid Color to remove any previously applied patterns, then click Apply.

Changing Chart Text

Versions automatically provides a top title and a legend for your chart. You may also provide left, right, and bottom titles, edit the title or legend text, and change text fonts for each text element on the chart.

To change the legend text:

1. Click Tools  .

   The Tools drop-down menu opens.

2. Select Data Editor from the menu bar.

   The Data Editor window appears in place of the chart. The Data Editor lists each value used to create the chart.

3. Edit the legend text as necessary.

   a. Double-click on a text element to be changed.

      The text is highlighted.

   b. Type the new text.

   c. Repeat steps a and b as often as necessary until you are finished editing data.

4. Click Tools  .

The Tools drop-down menu opens. There is a checkmark next to the Data Editor item.

5. Select Data Editor from the menu bar to remove the checkmark from the menu bar, close the Data Editor window, and regenerate the chart.

To add or change title text:

1. Click Edit Titles  .

   The Titles dialog opens.

2. Type the title text (or edit the existing text) for top, bottom, left, and/or right titles.

3. Click Apply to view your changes before closing the dialog or click OK.

To change font or color for any text element:

1. Click Change Text Fonts  .

   A drop-down menu listing all available text elements opens.

2. Select the text element for which to change fonts.

   A standard Font dialog opens.

3. Make changes to the font, style, point size, emphasis, and/or color, as appropriate.

Changing Viewing Options for 3D Charts

You can view charts in two dimensions (2D), three dimensions (3D), or full 3D view. In full 3D view, you can tilt and swivel the chart.

To switch between 2D and 3D:

- Click Switch Between 2D and 3D Views  .

To tilt and swivel a 3D chart:

1. Click Rotate Chart  .

   The 3D View Properties dialog opens.

2. Select the Full 3D View check box, if necessary, to activate full three-dimensional viewing options.

3. Using the mouse, drag the blue dot around the x-axis plane and/or drag the red dot around the y-axis plane.

Exploding a Pie Chart

To emphasize one or more values, a pie chart is frequently “exploded.” That is, one or more slices of the pie are shown separated from the rest.

To explode a pie chart:

1. Click on the slice to be separated from the rest of the pie.

   The mouse pointer changes to a cross.

2. Drag the slice away from the center of the pie chart.

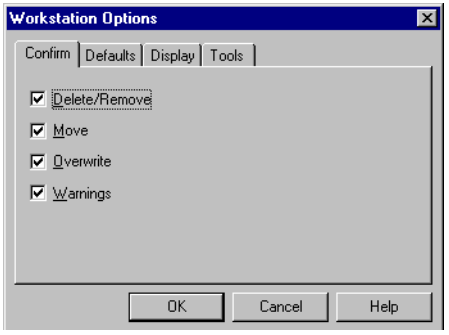
Chapter 10
Setting Workstation Options

Versions allows you and your team members to set workstation options that suit your individual work styles.

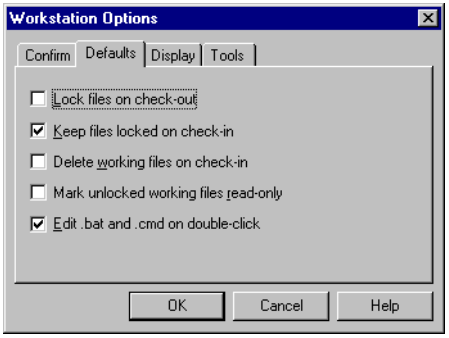
To customize Versions:

1. Select **Options** ⇨ **Workstation...** from the menu bar.

   The Workstation Options dialog opens.

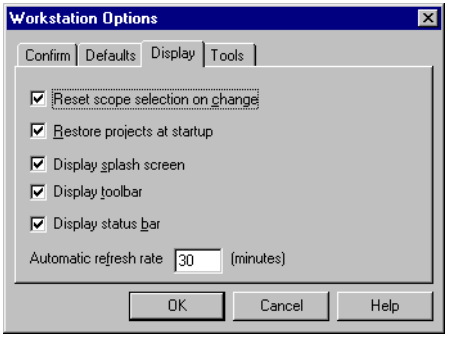


2. Use the Confirm tab to:
   - Confirm deletions, moves, and overwrites.
   - Display warnings.



3. Use the Defaults tab to:

   - Set up version control so that:
     - Files you check out are locked automatically.
     - Files you check in retain their locks.
     - Working files are deleted as you check in new versions.
     - Working files that are not locked by you become read-only as you check them out. Working files that you unlock become read-only as well.
   - Allow double-clicking to edit rather than execute .bat and .cmd files.

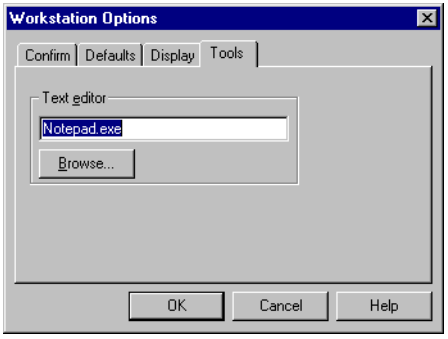


4. Use the Display tab to:

   - Automatically revert to the No Descendants scope button every time you change a tab or subproject. This saves you the time Versions would take to scan files and

so forth. Clear this check box so that only you change the scope buttons.

- Restore projects that were open when you last closed Versions.
- Display or hide the following:
  - Splash screen
  - Toolbar
  - Status bar
- Control the automatic refresh rate. Set a number of minutes from 1 to 90. To stop refreshing, use 0.



5. Use the Tools tab to:
   - Designate a text editor or word processor to be used when you double-click text files. The default editor is Windows NotePad.

Chapter 11
Maintaining Projects

As with all databases, the more you use project database files (especially for deletions), the more need there is to perform maintenance on them.

To start Project Maintenance:

- Double-click the Project Maintenance icon in the Versions program group.

  Or,

  Select **Project Maintenance** from the taskbar. By default, this would be **Start** ⇨ **Programs** ⇨ **Versions** ⇨ **Project Maintenance**.

  The Project Maintenance window opens.

  The window has both a toolbar and a status bar. You can enable or disable them using **View** ⇨ **Toolbar** and **View** ⇨ **Status Bar**. A checkmark appears in front of the menu item when the corresponding bar is enabled and disappears when the bar is disabled.

Repairing the Database

Use Repair when you cannot open the project database file, and the error “Unrecognized database format” is returned. Project Maintenance uses the database engine’s repair facility to fix the problem, but, depending on the file’s level of corruption, it may be unsuccessful.

To repair certain types of file corruption:

1. Select **Project** ⇨ **Repair...** from the menu bar.

   The Open dialog appears.

2. Select the project database file (.stp) to be repaired then click Open.

Packing the Database

As you delete the records for files, versions, and subprojects from a Versions project, the space they occupied needs to be reclaimed. Use the Pack command to eliminate space that is no longer used and to reduce the size of the project database file.

To pack the database:

1. Select **Project** ⇨ **Open...** from the menu bar.

   The Open dialog appears.

2. Select the project database file (.stp), then click Open.

   A project database window opens. Its title is the file you selected.

3. Select **Project** ⇨ **Pack** from the menu bar.

   Project Maintenance packs the project database file, eliminating the space formerly allocated to records that are now deleted. It also redisplays statistics in the window.

Validating the Database

Validating a database checks it for inconsistencies and verifies that project data relationships are in order. For example, if a subproject is supposed to have a child subproject, the validation checks to see if the child subproject really does exist. Problems with relationships may occur if the project suddenly shuts down in the middle of a critical operation.

To validate the database:

1. Select **Project** ⇨ **Open...** from the menu bar.

   The Open dialog appears.

2. Select a project database file (.stp), then click Open.

   A project database window opens. Its title is the file you selected.

3. Select **Project** ⇨ **Validate** from the menu bar.

   Project Maintenance validates the data in the project database file. It also redisplays statistics in the window.

Viewing the Log

When you use Project Maintenance commands to perform actions on a project database file (.stp), Project Maintenance logs those actions and the times at which they occur. The log file is a text file that has the same name as the project database file. However, its extension is .log.

To view the log:

1. Select **Project** ⇨ **Open...** from the menu bar.

   The Open dialog appears.

2. Select a project database file (.stp), then click Open.

   A project database window opens. Its title is the file you selected.

3. Select **Project** ⇨ **View Log** from the menu bar.

   Project Maintenance displays the log in Windows NotePad.

Chapter 12
Using Visual Diff

Visual Diff is a text comparison utility that is integrated into Versions. Using Visual Diff, you can quickly compare two different versions of a text file or two similar text files.

To access Visual Diff as a stand-alone utility:

1. Double-click the Visual Diff icon in the Versions program group.

   Or,

   Select **Visual Diff** from the taskbar. By default, this would be **Start** ⇨ **Programs** ⇨ **Versions** ⇨ **Visual Diff**.

   The Visual Diff window opens.

2. Select **File** ⇨ **Open...** from the menu bar.

   Or,

   Click Open  .

   The Select First File dialog opens.

3. Select the first text file to compare then click Open.

   The Select Second File dialog opens.

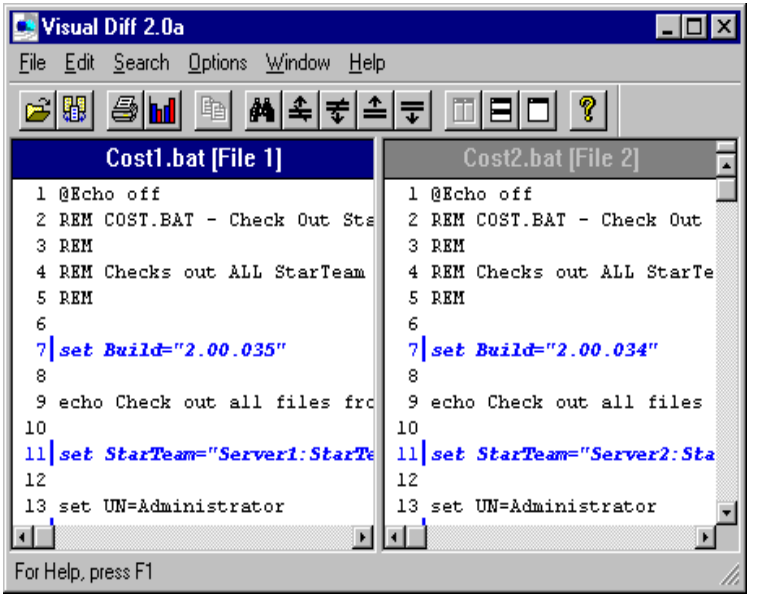
4. Select the second file to compare then click Open.

   Visual Diff compares the files and displays them in the Visual Diff main window.

NOTE
Visual Diff cannot compare files which exceed 32,768 lines of text.

Using the Visual Diff Main Window

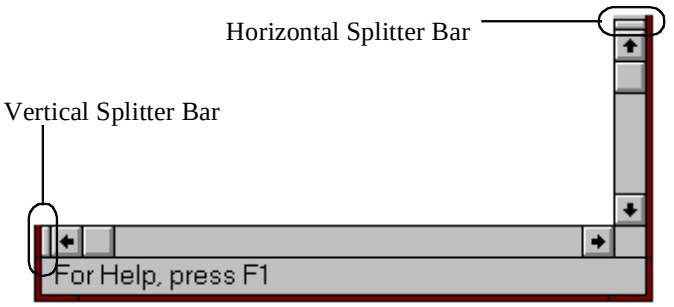
The Visual Diff main window consists of a menu bar, a tool bar, and an application workspace in which it displays the two files or versions being compared. You can display the compared files in two separate panes, either side-by-side or one on top of the other, or in merged view in a single-pane window.



Manipulating the Window Splitter Bars with the Mouse

In addition to using the Window menu options and the toolbar buttons to change the display options for the files, you can also manipulate the splitter bar with the mouse.

When in use, the splitter bar divides the Visual Diff main window. When not in use, the horizontal splitter bar is located above the vertical scroll bar and the vertical splitter bar is located to the left of the horizontal scroll bar.



Possible mouse operations include:

- Drag the splitter bar that separates the two panes to reproportion those panes.
- Drag the splitter bar that separates two panes to a window edge to remove the split and show the files together in a single pane.
- Drag the vertical splitter bar onto a window that is not split or is split horizontally to split the window vertically.
- Drag the horizontal splitter bar onto a window that is not split or is split vertically to split the window horizontally.
- Double-click on a splitter bar that is not in use to activate it.
- Double-click on a splitter bar that is in use to deactivate (remove) it.

Comparing Files and Viewing Differences

Whether you launch Visual Diff from within Versions or open it as a stand-alone application, Visual Diff automatically compares the two files or versions as soon as you identify them. The first file that you select appears on the left, and the second file appears on the right. If you access Visual Diff from within Versions to compare two versions of the same file, the older file is on the left and the newer file is on the right.

If there are no differences between the files, Visual Diff displays a message stating that the files are the same and does not display them.

Once Visual Diff compares and displays the files, the differences are shown in color. The default colors are as follows:

Black
Matching lines are displayed in black.

Red
Deleted lines appear in red and struck through. Deleted text is text contained in the file on the left only.

Green
Inserted lines appear in green. Inserted text is text contained in the file on the right only.

Blue
Changed lines appear in blue. Lines that are similar but not identical are considered to be changed.

When the compared files are shown in a single pane, matching, deleted, and inserted lines are shown only once in their appropriate color. Changed lines appear twice (in blue). The first line is from the file selected first or the older version and is struck through. The second line is from the file selected second or the newer version.

To change the colors for matching, deleted, inserted, and/or changed text:

1. Select **Options** ⇨ **Workstation** from the menu bar.

2. Select the Color tab.

3. Select the kind of text to be changed from the list, then click Select Color....

4. Select one of the predefined colors or click Define Custom Colors>> and define a color to use for the text element using the standard Windows color dialog.

To return to the default Visual Diff colors:

1. Select **Options** ⇨ **Workstation** from the menu bar.

2. Select the Color tab.

3. Click Default Colors.

Searching for Text

You can locate text strings in the active pane.

To find text:

1. Select **Search** ⇨ **Find...** from the menu bar.

   Or,

   Click Find  .

   The Find dialog opens.

2. Type the string to be located in the Find What text box.

3. If you want to locate the text exactly as you typed it with regard to case, select the Match Case check box.

4. If necessary, specify a direction for the search.

   Visual Diff remembers the search string until you enter another search string or until you close the application. To locate the next instance of the text, use the Find Next option. (Change panes if you want to search the other pane.)

You can also search for the next set of matching or non-matching lines or the previous set of matching or non-matching lines using the Search menu or the toolbar buttons. Visual Diff locates the lines and positions them at the top of the panes.

Merging or Printing Compared Files

When you merge two files or versions to create a new file, the system uses a series of characters to delineate text rather than using colors. Once the files are merged, you must delete extraneous text manually.

Key to Reading Merged File:

#######	Separates changed lines from the rest of the text. Both versions of the changed lines are shown, separated by **~~~~~~~**.
- - - - - - -	Separates deleted lines from the rest of the text
+++++++	Separates inserted lines from the rest of the text

Below is an example of a merged file.

```
@ECHO OFF
C:\WINDOWS\NET START
######################### 1 Line(s) Changed ##########
C:\DOS\SMARTDRV.EXE /X
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
C:\DOS\SMARTDRV.EXE 1024 /X
################################################
PROMPT $p$g
PATH c:\windows;c:\dos;c:\mach32
----------- 1 Line(s) Deleted --------------------
SET DIRCMD=/oen
------------------------------------------------
REM Set location for temp files.
######################### 1 Line(s) Changed ##########
SET TEMP=C:\DOS
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
SET TEMP=C:\TEMP
################################################
++++++++++++++++++++++++ 1 Line(s) Inserted +++++++++
SET TMP=C:\TEMP
++++++++++++++++++++++++++++++++++++++++++++++++
```

To merge compared files:

1. Open and compare the files.

2. Select **File** ⇨ **Merge Files...** from the menu bar.

   Or,

   Click Merge  .

   The Save Merged Comparison As dialog opens.

3. Select a directory in which to store the file, name the file, and click Save.

   Visual Diff creates the new file and, if the editing option is set, opens it in Windows NotePad or the text editor of your choice.

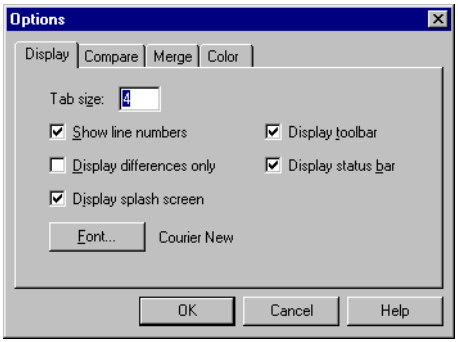
Setting Options

You can customize Visual Diff to suit your individual preferences and work style by setting display, comparison, merge, and color options.

To customize Visual Diff:

1. Select **Options** ⇨ **Workstation...** from the menu bar.

   The Options dialog opens.



2. Select one of the tabs at the top of the dialog.

   - Select **Display** to change the display of text in the compared files or versions. You can change the font and tab settings, display only the differences, and turn line numbers on or off.

     You also use this option to turn off the Visual Diff splash screen which appears at start up and display or hide the toolbar and status bar.
   - Select **Compare** to set case and whitespace options during a search.
   - Select **Merge** to determine when to view a merged file and with what editor.
   - Select **Color** to change the default colors for deleted, changed, inserted, and matching text.

Chapter 13
Using the Command-line Interface

You can perform the following operations from a command-line session using the command stcmd and the appropriate options. These commands also allow you to perform Versions's version control operations from any development environment that allows you to add tools to menus.

- Add files (**stcmd add**)
- Check in files (**stcmd ci**)
- Check out files (**stcmd co**)
- Lock or unlock files (**stcmd lck**)
- Compress or decompress files (**stcmd cmp**)
- Merge files (**stcmd mrg**)
- Designate build (**stcmd dbd**)
- Designate milestone (**stcmd dms**)
- Change description (**stcmd dsc**)
- Set maximum number of versions (**stcmd mnv**)
- Give versions permanent or non-permanent status (**stcmd prm**)
- Delete versions (**stcmd dvr**)
- Purge non-permanent versions (**stcmd prg**)

Use **/?** to display the options available with the command.

For example, **stcmd /?** displays the general syntax for stcmd and **stcmd add /?** displays the options available when adding files. (Using stcmd and stcmd add without the /? also works.)

Syntax

The syntax for the command line uses the following conventions:

Bold Information that you must use exactly as shown (if you use it).

Italics Information that you replace with the names of your files, subprojects, etc.

[ ] Optional information appears in square brackets.

Common Parameters

Some parameters show up in all or almost all of the commands explained in this chapter. In each command, they have exactly the same meaning so they are explained in this section and not repeated later. Parameters that do not appear in all commands or vary in meaning from command to command are explained with the commands.

/p Indicates what project is to be used by providing one of the following:

- The name of the project database file for the project. Use the complete pathname to the file and the .stp extension in double quotation marks. For example, “c:\stardraw.stp”.·
- The server description and project name combination. Separate the two with a colon and delimit them with double quotation marks. For example, “Orion:NEF”, where Orion is the server name and NEF is the name of the project. These are case insensitive.

/s Provides the name of the subproject, preceded by the names of any other subprojects that separate it from the root subproject. (Do not include the name of the root.) The names of each subproject in the "path" are separated by exclamation points (!) and the complete string is enclosed in double quotation marks. When /s is not used, the files are assumed to be in or intended for the root subproject.

/is Applies the command to one of the following:

- All the subprojects which are children of the subproject specified with the /s option
- All the subprojects in the project when the /s option is not used

/un Provides the name of the user performing the action in double quotation marks. Without this option, the current network logon name is used.

/pw Provides the password (in double quotation marks) for the user specified with the /un option. This option is required only when project security is enabled and the password is not blank.

/q Suppresses progress reporting. Without this option, messages about each action appear on the screen as the action is performed.

/x Bypasses error messages. Without this option, you will have to confirm error messages.

files... Specifies the files to be used in the command by name or by file specification (such as *.c). Without this option, the default is *.*. This must always be the last parameter. Any parameters after it are ignored. File specifications that include spaces must be delimited by double quotation marks.

Adding Files: stcmd add

Use stcmd add to add files to a project from the command line.

Syntax

stcmd add [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] [/l | /u [/ro]]**
[/d "*fileDescription***"] [/vl "***versionLabel***"] [/k]**
[/md ***maxDelta*****] [/mo** ***maxOmega*****] [/fo] [***files...***]**

/l Locks the files (or keeps them locked) after this action is performed. The locker is the user specified with the /un option. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed. When neither /l nor /u are used, the files are locked by default.

/u Unlocks the files (or leaves them unlocked) after this action is performed. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed.

/ro Used with /u to make the unlocked working file read-only after this operation is completed. Without this option, the file is read/write. Use /ro to make it less likely that you will edit a file that is not locked by you.

/d Sets the description for the specified files to the specified description. The description is enclosed in double quotation marks.

/vl Provides a version label. The label is enclosed in double quotation marks.

When adding files, this label becomes the label for the versions of the files. Without this option, the versions are added to the project without labels.

/k Makes the version being added a permanent version. Without this option, the version is non-permanent.

/md Specifies the maximum number of versions to be stored for the files using delta storage. Without this option, the maximum number is Maximum.

/mo Specifies the maximum number of versions to be stored for the files using omega storage. Without this option, the maximum number is 10.

/fo Forces text files to be stored using omega storage. Without this option, delta storage is used for text files. (Binary files are always stored using omega storage, regardless of this option.) Once a text file has been stored using omega storage, it will always be stored using omega storage.

Example

The following example uses stcmd add to add .doc files to a subproject of StarDraw called User Manual. It locks the files and gives them the description “First draft of chapter”.

```
stcmd add /p “c:\stardraw.stp” /s “User Manual” /un “NAllis”
/pw “password” /l /d “First draft of chapter” *.doc
```

Checking in Files: stcmd ci

Use stcmd ci to check in project files from the command line.

Syntax

```
stcmd ci [ /p ”projFile.stp” | “serverDesc:projName”]
[/s “subproj1!subproj2!...”]  [/is]  [/un “userName”]
[/pw “password”]  [/q]  [/x]  [/l | /u [/ro]]
[/vl “versionLabel”]  [/k]  [/f NCI]  [/o]  [/fo]
[/r “reasonCheckIn”]  [files...]
```

/l Locks the files after this action is performed. The locker is the user specified with the /un option. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed.

/u Unlocks the files (or leaves them unlocked) after this action is performed. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed.

/ro Used with /u to make the unlocked working file read-only after this operation is completed. Without this option, the file is read/write. Use /ro to make it less likely that you will edit a file that is not locked by you.

/vl Provides a version label. The label is enclosed in double quotation marks.

When checking in files, this label becomes the label for the versions of the files. Without this option, the versions are checked in without labels.

/k Makes the version being checked in a permanent version. Without this option, the version is non-permanent.

/f NCI Specifies the check in of all the files that need to be checked in (i.e., all the files that match the “Files to Check In” filter). When you use this option, any file specifications at the end of the command are ignored. When neither /f NCI nor Files... is used, the default is *.*.

/o Allows you to check in files that are identical to the most recently checked in versions.

Without this option, identical files are not processed, and an error occurs.

/fo Forces text files to be stored using omega storage. Without this option, delta storage is used for text files. (Binary files are always stored using omega storage, regardless of this option.) Once a text file has been stored using omega storage, it will always be stored using omega storage.

/r Provides a reason for checking in the files. The reason is enclosed in double quotation marks. Without this option, no reason is stored for the checked-in versions.

Example

The following example uses stcmd ci to check in .bmp files to a subproject called On-line Help. On-line Help is a subproject of SourceCode, which is a subproject of StarDraw. The command unlocks the files, makes the working copy read-only, and gives them a reason for check in.

```
stcmd ci /p “c:\stardraw.stp” /s “SourceCode!On-line Help”
/un “NAllis” /pw “password” /u /ro /r “revised for beta”
*.bmp
```

Compressing and Decompressing Files: stcmd cmp

Use stcmd cmp to compress or decompress project vault files from the command line.

Syntax

stcmd cmp [**/p** ”*projFile.stp*” **|** “*serverDesc:projName*”**]**
[/s “*subproj1!subproj2!...*”**]** **[/is]** **[/un** “*userName*”**]**
[/pw “*password*”**]** **[/q]** **[/x]** **[/c | /dc]** **[***files...***]**

/c Compresses the files.

/dc Decompresses the files.

Example

The following example uses stcmd cmp to compress all the project vault files in SourceCode, a subproject in the project StarDraw, as well as all the files in subprojects of SourceCode.

```
stcmd cmp /p "c:\stardraw.stp" /s "SourceCode" /is
/un "NAllis" /pw "password" /c *.*
```

Checking Out Files: stcmd co

Use stcmd co to check out project files from the command line.

Syntax

stcmd co [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] [/l | /u [/ro]]**
[/vn ***versionNumber*** **| /vl "***versionLabel***" | /vd** ***asOfDateTime*****]**
[/f NCO] [/o] [/ts] [*files...***]**

/l Locks the files (or keeps them locked) after this action is performed. The locker is the user specified with the /un option. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed.

/u Unlocks the files (or leaves them unlocked) after this action is performed. When neither /l nor /u is specified, Versions leaves the lock status of each file as it was before the command was executed.

/ro Used with /u to make the unlocked working file read-only after this operation is completed. Without this option, the file is read/write. Use /ro to make it less likely that you will edit a file that is not locked for you.

/vn Specifies the version number to be checked out for each of the files specified by this command. Without this option or /vl or /vd, the most recent version of the file is checked out.

/vl Provides a version label. The label is enclosed in double quotation marks.

The label is used to identify the versions to be checked out. Without this option or /vn or /vd, the most recent version of the file is checked out.

/vd Specifies a date and optionally a time to be used to identify versions of the specified files. The last version checked in before the specified date is the version checked out for each file.

The date can be specified as mm/dd/yy or mm/dd/yyyy. The time can be specified in hours; hours and minutes; or hours, minutes, and seconds (separated by colons). The syntax for the time is: [hh [:mm [:ss]]]

When specifying both a date and a time, separate them with a space and use double quotation marks as in “11/13/96 12:01:01”.

Without this option or /vl or /vn, the most recent version of the file is checked out.

/f NCO Specifies the check out of all the files that need to be checked out (i.e., all the files that match the “Files to Check Out” filter). When you use this option, any file specifications at the end of the command are ignored. When neither /f NCO nor Files... is used, the default is *.*.

/o When checking out files, forces the check out when the working file appears to be more recent.

Without this option, more recent working files are not processed, and an error occurs.

/ts Sets each working file´s time stamp to the check-out time. Without this option, the time stamp is the same as for the checked in version that is being copied to the working directory.

Example

The following example uses stcmd co to lock and check out .doc files from a subproject of StarDraw called User Manual.

```
stcmd co /p "c:\stardraw.stp" /s "User Manual" /un "NAllis"
/pw "password" /l *.doc
```

Designating Builds: stcmd dbd

Use stcmd dbd to designate a project build from the command line. A build includes all the files in the project.

When project security is enabled, only an administrator can designate a build.

Syntax

```
stcmd dbd [ /p "projFile.stp" | "serverDesc:projName"]
[/un "userName"]  [/pw "password"]  [/q]  [/x]
/nl "buildLabel"  [/vd asOfDateTime]
```

/nl Specifies the build label. The label is enclosed in double quotation marks. This option is required.

/vd Specifies a date and optionally a time to be used to identify the versions to be included in the build. The last version checked in before the specified date is the version used for each file.

The date can be specified as mm/dd/yy or mm/dd/yyyy. The time can be specified in hours; hours and minutes; or hours, minutes, and seconds (separated by colons). The syntax for the time is: [hh [:mm [:ss]]]

When specifying both a date and a time, separate them with a space and use double quotation marks as in “11/13/96 12:01:01”.

Without this option, the most recent versions of the files are included.

Example

The following example uses stcmd dbd to designate as a build the most recent versions of all the files in the project StarDraw.

```
stcmd dbd /p “c:\stardraw.stp” /un “NAllis” /pw “password”
/nl “Build 14”
```

Designating Milestones: stcmd dms

Use stcmd dms to designate a project milestone from the command line. A milestone includes all the files in the project.

When project security is enabled, only an administrator can designate a milestone.

Syntax

```
stcmd dms [ /p ”projFile.stp” | “serverDesc:projName”]
[/un “userName”]  [/pw “password”]  [/q]  [/x]
/nl “milestoneLabel”  [/vl “buildLabel” | /vd asOfDateTime]
```

/nl Specifies the milestone label. The label is enclosed in double quotation marks. This option is required.

/vl Provides a build label. The label is enclosed in double quotation marks.

The label is used to identify the versions to be included in the milestone. Without this option or /vd, the most recent versions of the files are included.

/vd Specifies a date and optionally a time to be used to identify the versions to be included in the milestone. The last version checked in before the specified date is the version used for each file.

The date can be specified as mm/dd/yy or mm/dd/yyyy. The time can be specified in hours; hours and minutes; or hours, minutes, and seconds (separated by colons). The syntax for the time is: [hh [:mm [:ss]]]

When specifying both a date and a time, separate them with a space and use double quotation marks as in “11/13/96 12:01:01”.

Without this option or /vl, the most recent versions of the files are included.

Example

The following example uses stcmd dms to designate as a milestone the most recent versions of all the files in the project StarDraw.

```
stcmd dms /p “c:\stardraw.stp” /un “NAllis” /pw “password”
/nl “Beta”
```

Changing Descriptions: stcmd dsc

Use stcmd dsc to change a file description from the command line.

Syntax

stcmd dsc [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] /d "***newDescription***" [***files...***]**

/d Provides a description for the files. The description is enclosed in double quotation marks. Without this option, an error message occurs.

Example

The following example uses stcmd dsc to change the description of stdafx.cpp in SourceCode, a subproject in the project StarDraw.

```
stcmd dsc /p "c:\stardraw.stp" /s "SourceCode" /un "NAllis"
/pw "password" /d "Source code for Project" stdafx.cpp
```

Deleting Versions: stcmd dvr

Use stcmd dvr to delete versions of project files from the command line.

When project security is enabled, only an administrator can delete versions.

Syntax

stcmd dvr [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] [/o]**
[/vn ***versionNumber*** **| /vl "***versionLabel***"][***files...***]**

/o Forces the specified versions to be deleted even if they are permanent. In addition, you must confirm all the deletions of permanent files.

Without this option, only nonpermanent versions are deleted.

/vn Specifies the version number to be deleted for each of the files specified by this command. Without this option or /vl, the most recent version of the file is deleted.

/vl Provides a version label. The label is enclosed in double quotation marks. The label identifies the versions to be deleted. Without this option or /vn, the most recent version of the file is deleted.

Example

The following example uses stcmd dvr to delete the most recent versions of all the files in SourceCode, a subproject in the project StarDraw, as well as all the files in subprojects of SourceCode.

```
stcmd dvr /p “c:\stardraw.stp” /s “SourceCode” /is
/un “NAllis” /pw “password” *.*
```

Locking and Unlocking Files: stcmd lck

Use stcmd lck to lock or unlock project files from the command line.

Syntax

stcmd lck [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] [/l | /u [/ro]] [***files...***]**

/l Locks the files.

/u Unlocks the files.

/ro Used with /u to make the unlocked working file read-only after this operation is completed. Without this option, the file is read/write. Use /ro to make it less likely that you will edit a file that is not locked by you.

Example

The following example uses stcmd lck to unlock all the files in SourceCode, a subproject in the project StarDraw, as well as all the files in subprojects of SourceCode.

```
stcmd lck /p "c:\stardraw.stp" /s "SourceCode" /is
/un "NAllis" /pw "password" /u *.*
```

Setting Maximum Numbers of Versions: stcmd mnv

Use stcmd mnv to set the maximum number of versions for project files from the command line.

When project security is enabled, only an administrator can set these maximums.

Syntax

stcmd mnv [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x] [/md** ***maxDelta*****] [/mo** ***maxOmega*****]**
[f*iles...***]**

/md Specifies the maximum number of versions to be stored for the files using delta storage. /md and/or /mo must be used in this command. You can use both, but you cannot use neither.

/mo Specifies the maximum number of versions to be stored for the files using omega storage. See /md for more information.

Example

The following example uses stcmd mnv to set the maximum number of versions to 20 for delta files in SourceCode, a subproject in the project StarDraw, as well as all the files in subprojects of SourceCode.

```
stcmd mnv /p "c:\stardraw.stp" /s "SourceCode" /is
/un "NAllis" /pw "password" /md 20 *.*
```

Merging Files: stcmd mrg

Use stcmd mrg to merge one or more files from a branch subproject back into its root subproject. For each file, Versions performs a three-way merge and copies the resulting file to the working directory for the root subproject, overwriting the working file that is there. The three file versions that are merged are:·

- The latest checked-in version of the specified file in the branch subproject·
- The latest checked-in version of the file in the root subproject that corresponds to the file in the branch·

- The version of the file in the root subproject from which the file in the branch was derived

When you branch a subproject, the latest version of each file in the root subproject is copied to the branch. Initially these two files are identical. However, both can be checked in and out and have their contents modified. Eventually they are simply two files with the same name and the same parent version (the version of the file originally copied from the root subproject to the branch). Merging these three versions incorporates all the changes made in the root and the branch subproject. You can edit the merged file and then check it in to the root subproject.

Syntax

stcmd mrg [/p "*projFile.stp***" | "***serverDesc:projName***"] [/s "***subproj1!subproj2!...***"] [/is] [/un "***userName***"] [/pw "***password***"] [/q] [/x] [files...]**

Example

The following example uses stcmd mrg to merge the file ipframe.cpp from the branch subproject SourceCode RSL with its "sister file" in the root subproject SourceCode. Versions knows that the SourceCode RSL subproject was branched from SourceCode as well as what version of ipframe.cpp in SourceCode became the first version of ipframe.cpp in SourceCode RSL. Versions merges the current versions of ipframe.cpp found in both the root and branch subproject with the parent version from the root subproject. The results are copied to the working directory for SourceCode as the working file ipframe.cpp.

```
stcmd mrg /p "c:\stardraw.stp" /s "SourceCode RSL" /un "NAllis" /pw "password" ipframe.cpp
```

Changing Version Status: stcmd prm

Use stcmd prm to make the specified file versions permanent or non-permanent.

Syntax

stcmd prm [/p "*projFile.stp***" | "***serverDesc:projName***"]**
[/s "*subproj1!subproj2!...***"] [/is] [/un "***userName***"]**
[/pw "*password***"] [/q] [/x]**
[/vn ***versionNumber*** **| /vl "***versionLabel***" | /vd** ***asOfDateTime***]
[/k] [*files...***]**

/vn Specifies the version number to be used for each of the files specified by this command. Without this option or /vl or /vd, the most recent version of the file is used.

/vl Provides a version label. The label is enclosed in double quotation marks.

The label is used to identify the versions to be used. Without this option or /vn or /vd, the most recent version of the file is used.

/vd Specifies a date and optionally a time to be used to identify versions of the specified files. The last version checked in before the specified date is the version used for each file.

The date can be specified as mm/dd/yy or mm/dd/yyyy. The time can be specified in hours; hours and minutes; or hours, minutes, and seconds (separated by colons). The syntax for the time is: [hh [:mm [:ss]]]

When specifying both a date and a time, separate them with a space and use double quotation marks as in "11/13/96 12:01:01".

Without this option or /vl or /vn, the most recent version of the file is used.

/k Makes the specified versions permanent versions. Without this option, the version becomes non-permanent.

Example

The following example uses stcmd prm to make permanent the most recent versions of all the files in SourceCode, a subproject in the project StarDraw, as well as all the files in subprojects of SourceCode.

```
stcmd prm /p “c:\stardraw.stp” /s “SourceCode” /is /un “NAl-
lis” /pw “password” /k *.*
```

Purging Versions from Files: stcmd prg

Use stcmd prg to purge nonpermanent file versions from the command line.

When project security is enabled, only an administrator can purge files.

Syntax

stcmd prg [/p ”*projFile.stp***” | “***serverDesc:projName***”]**
[/s “*subproj1!subproj2!...***”] [/is] [/un “***userName***”]**
[/pw “*password***”] [/q] [/x] [***files...***]**

Example

The following example uses stcmd prg to purge all the non-permanent versions of the files in SourceCode, a subproject in the project StarDraw, as well as all the non-permanent versions of the files in subprojects of SourceCode.

```
stcmd prg /p “c:\stardraw.stp” /s “SourceCode” /is
/un “NAllis” /pw “password” *.*
```

Chapter 14
Importing Projects from PVCS

Versions can import projects from the PVCS 32-bit edition of 5.2.1, allowing you to update your current project development environment to Versions while maintaining version history information.

The PVCS Import tool is a stand-alone executable, pvcsimp.exe. It locates a PVCS installation by looking for PVCS's islv.ini file in your Windows directory (for example, c:\windows\islv.ini). It displays a list of projects from the .ini file and allows you to import projects into Versions. Importing PVCS projects does not alter or delete those PVCS projects. The import process only copies information from them into Versions databases and vaults, creating Versions projects.

PVCS Import Features

- Reads the PVCS project information and allows you to select PVCS projects for import.
- Maps PVCS folders to Versions subprojects.
- Imports all or only the last revision of the main trunk of all archives.
- Imports all check-in information such as author, (original) date of check-in, version comment, labels, etc.
- Reports all actions performed during the import online in the Versions PVCS Import dialog and logs the actions in a log file as well.
- If you cancel the import after it starts, the partially constructed project is deleted. So are all the vault files and databases. The directory structure is left unchanged. The original PVCS files are left undeleted.

Limitations and Differences

- Imports projects from the PVCS 32-bit edition of 5.2.1 only.
- Does not support multiple references to a vault (archive) file. When the Project Folder and other user-created PVCS folders contain the same file, that file is imported as a part of only one of those user-created folders.
- Does not support PVCS branches. Only the main trunk is imported. Ignored branches are not reported in the log file.
- Does not import promotion definitions or private folders because Versions does not support them.
- PVCS uses revision names such as “1.1”, while Versions uses version numbers, such as 1, 2, and 3. The revision numbers of the PVCS main trunk are mapped to Versions version numbers.

Importing Projects from PVCS

You can import projects (one at a time) from PVCS 5.2 to Versions. Be sure to check in all changed working files prior to performing the import so that no work is lost.

To import projects from PVCS:

1. Select PVCS Import to run pvcsimp.exe. (In Windows 95, PVCS Import is on the submenu for Versions. It is an icon in the Versions program group in NT.)

   If PVCS Import finds a PVCS installation, the Versions PVCS Import dialog opens.

2. Select a project to import from the Projects list box.

3. To see details about the PVCS project, click Project Detail.
   The Log Output box displays information about the selected project, such as its name, configuration file, project descriptor root, archive directory, working directory, folder names, and filenames.

4. Click Import... to specify the information Versions needs to create a project. The Project Import Parameters dialog opens.

5. Type a name for this project in the Project Name text box (or use its PVCS name which is the default).

6. Type the pathname for a working directory for the project.

   Or,

   Click Browse... to locate one.

7. Type the pathname for the file that will store the Versions project database file (.stp).

   The directory that stores the project database file will also store the project vault, the equivalent of PVCS's archive directory.

   The revisions (or versions) of files from the imported project will go into the project vault. Versions uses the name of the project database file (but this time with the extension .vlt) as the name of the vault directory. Let Versions control these files! Do not delete them or modify them in any way.

8. Type the name of a person who will be an administrator for the project in the First User text box.

   An administrator can set project properties, assign users, etc. Authors of files in the project also become administrators as part of the import process.

9. Select Import All Revisions (main trunk) or the Import Latest Revision Only option button to indicate how many versions of each file to import.

10. Select the Compress check box to compress the vault files of the imported file or leave it cleared to avoid compression.

11. Use the default log file name to record the actions performed during the import process.

    Or,

    Type the pathname for a log file.

    Or,

    Click Browse... to locate one.

12. Click Import.

    A progress dialog displays a progress meter. A Cancel button in the progress dialog allows you to cancel the import.

13. After the import, the Versions PVCS Import dialog reappears.
    Repeat steps 2 through 11 to import another project or click Done.

Inspecting the Imported Files

After you import a project from PVCS 5.2 to Versions, you will want to look at the results.

To inspect the imported files:

1. Run Versions.
2. Click Open Project  and locate the .stp file for the imported project.

   The project window opens.
3. Click All Descendants  to see the contents of all the subprojects.

   You see the name of each file, its author, description, subproject, path, etc.
4. Use Versions Help for more details.

Checking Out the Imported Files

After you import a project from PVCS 5.2 to Versions, it is a good idea to check out all of the files you will be using.

To check out all the imported files:

1. Run Versions.
2. Click Open Project  and locate the .stp file for the imported project.

   The project window opens.
3. Click All Descendants  to see the contents of all the subprojects.

   You see the name of each file, its author, description, subproject, path, etc.
4. Select **Files** ⇨ **Select All...** from the menu bar.
5. Select **Files** ⇨ **Check Out...** from the menu bar. Select the Force Check Out check box, and click OK.

Specifying Project Properties

After you import a project from PVCS 5.2 to Versions, you will want to set properties for it in Versions.

To set project properties:

1. Run Versions.
2. Open the imported project.
3. Select **Project** ⇨ **Properties...** from the menu bar.

   The Project Properties dialog opens.
4. Use Versions Help for more details.

Appendix A Glossary

add files
The process of placing files under version control by adding them to a Versions project.

administrator
A functional role in the operation of Versions. To become a project administrator, you must create the project, import the project, or have another administrator change your user properties to designate you as an administrator.

When project security is enabled, only project administrators can create or branch subprojects, change project properties, break locks, manage team member information, modify the number of versions for a file, designate or delete permanent versions, purge non-permanent versions, designate builds and milestones, and perform delete operations.

alphanumeric
A value consisting of the letters A-Z, and the digits 0-9.

archive
For PVCS users, a synonym for vault. See vault.

Others often use it to mean a backup copy of a program or data file saved to ensure against loss in case the original materials are deleted or damaged. Archived files can be stored on tape, floppy disks, or another computer system.

audit entry
A record of an action performed on a Versions project that appears in the audit log. For example, every time a file is added to a project, that action is entered in the audit log.

audit log

A chronological record kept by Versions showing all actions performed on Versions projects, subprojects, files, and users.

author

Information stored by Versions about files and versions.

For a file: The user who added the file to the project.

For a version: The user who checked in the version.

automatic refresh

An automatic update feature of Versions. Since projects, subprojects, and files managed by Versions are subject to continuous revision by various team members, screen information can become outdated after it is first displayed. The Automatic Refresh function periodically reads the project database to update the displayed information.

binary file

Any file that is not strictly an ASCII text file. Binary files contain programs or data in machine code. Examples of binary files and their respective file name extensions are executables (.exe), resources (.res), word processing documents (.doc in Word for Windows), spreadsheets (.xls in Excel), object files (.obj) and bitmaps (.bmp).

branch

The process of creating a second, independent subproject from the contents of an existing subproject. The branch may become a project in its own right or later be merged back into the subproject from which it originated. For example, the development of a product for a new operating system may start with the existing files for the first operating system as its base.

Also a branch of a tree, such as the project tree or a topic tree.

build

The process of compiling, assembling, and linking all project files in proper sequence to produce the project´s product. Versions uses the build command provided as a project or subproject property to create the build.

Also an event in the life cycle of a product chosen to represent a quantifiable step in progress for a project. For example, a software product may deliver a new build every few days to the team members who test the product. Designating a build creates a set of non-permanent versions at the project level. Each version of a file in the build uses the name of the build as its label. When project security is enabled, only an administrator can designate a build. A build cannot be designated at the subproject level.

build command

A command that you specify so that Versions can, with a single operation, compile, assemble, and link all project files in proper sequence to produce the project or subproject's product.

build label

The name or number given to a specific build. When you designate a build, Versions assigns the build label to the specified version of each file in the project.

check-in

The operation performed on a file that has undergone revision that stores the new file version in the Versions project vault.

Versions permits a number of individuals to work on a common set of files by allowing only one team member to revise a project file at a time. Check-in marks the end of one revision. The team member who checks in the file can keep it locked or release the file to others by unlocking it.

check-out

The operation that copies a version of a file from the Versions project vault to a team member´s working directory. A team member can check out a file with or without the intention to alter that file. Versions permits a number of individuals to work on a common set of files by allowing only one team member to revise a project file at a time. Locking the file marks the beginning of one author's revision.

column header

The label or name for a column in lists that appear in the right pane of the Versions project window. Clicking the header initiates sort operations based on the values in that column.

To hide a column, locate the bar that follows the column header (and separates that header from the one that follows it). Drag the bar to the left edge of the column. When you release the mouse button, you should no longer see the header.

command-line file

When you compare two files or versions, Versions sends a command-line file to Visual Diff containing the appropriate parameters. If there is no space for this file or no temp directory to create it in, an error message appears.

context menu (or pop-up menu)

A menu that appears when you right-click an item in the project window. The menu contents vary with the context, *i.e.* what has been clicked. In general, a context menu contains the most commonly used items from a related pull-down menu on the menu bar. In some cases, the items on the context menu are shortcuts. For example, if you select Check In from the context menu that appears in the files list, you bypass the Check In dialog.

defect

A fault, error, or suggestion for a product. If you use Versions along with StarTeam Workstation 2.0, you can check-in files and fix defects at the same time.

delete

To remove information from Versions databases. You can delete:

- Projects. Deleting a project deletes the project database file (.stp) and the vault for that project.
- Project files and working files.
- Versions of a file. To delete a file version, the file must be locked.
- Audit entries. You can clear the entire log or delete everything up to a specified date.
- Working files as part of removing a subproject or file from version control.

When project security is enabled, only an administrator can perform these operations.

See also remove from version control.

delta storage

A method of computing differences between progressive versions of a file.

Versions implements reverse deltas for text files, which means that only the latest version of the file is saved in its entirety. Delta storage reduces the disk space necessary to store the version information of a file by comparing new and previous versions of the file and storing only the differences.

e-mail

An organized system for delivery of paperless, “electronic” messages, named for its similarities to the postal mail system. Versions provides advantages (such as mailing files and reports) for teams that use a MAPI-compliant e-mail system.

file compression

A technique for reducing the size of a file by removing redundant information from it. Most disk files contain repetitions of common sequences of characters. Compression algorithms remove the additional occurrences of these sequences and save information that permits their restoration. By selecting compression of files, you may reduce vault space requirements as well as improve performance.

files list The list of files for the subproject, selected from the project tree, that is displayed when you select the Files tab. The list is further refined by the scope button and filter you select.

filter The criteria used to select a few items from among many. The Filter drop-down list box in the project window allows you to display only the files or audit entries that are of interest.

filtering The technique of selecting a few items from among many by specifying criteria that eliminate unwanted items. For example, you can select only files submitted within the last month.

group band The label that separates data with the same value in a given column from the data above it in the list. Depending on the column, group bands may appear in the project window after you click a column header to sort the data based on the value in that column. For example, sorting by the Author or User column breaks down the information by team member and displays group bands to make viewing and report generation easier.

history list The list of versions for the file selected from the project tree that is displayed when you select the History tab from the bottom of the project window's right pane or from the File Properties dialog.

keyword Reserved words beginning and ending with a dollar sign ($). When used in a text file, Versions replaces them with the data that they represent. For example $Author$ is replaced by the name of the user who checked in the file.

keyword expansion A technique used to insert information in a text file in which keywords are replaced by the data they represent. For example $Author$ is replaced by the name of the user who checked in the file.

lock

The process by which you indicate to others that you are revising a file. Others can check out a file you have locked, but they cannot check it in unless they break your lock. When security is enabled, only administrators can break locks.

Optionally, team members can have unlocked files marked read-only, which prevents them from making any changes to files that may be locked by others.

A file must be locked before you can check it in, delete it or one of its versions, change its number of maximum non-permanent versions, purge its non-permanent versions, or change a version from non-permanent to permanent or vice versa. When security is enabled, you must be an administrator (and lock the file) to perform any of the above except check-in.

locker

The team member who has locked a file is known as its locker.

MAPI

Acronym from (Mail Application Programming Interface). A programming interface that permits an application to send and receive electronic mail via the Microsoft Mail messaging system.

merge

The process of combining a working file with the latest checked-in version of that file or of combining a branched file with the root file from which it was branched. In either case, the combined file is placed in the user´s working directory where it can be inspected or revised before being checked in.

milestone

An event in the life cycle of a product chosen to represent a significant step in progress, for example, the alpha, beta, or final release of a product. In Versions, a milestone designates a single set of permanent versions for the files in a project.

Each version in the milestone has the same label. When project security is enabled, only an administrator can designate a milestone. Milestones cannot be designated at the subproject level.

NetBIOS over NetBEUI

A common network protocol for PC LANs that provides session and transfer services. NetBEUI is an acronym for NetBIOS Extended User Interface and provides a standardized transport frame for NetBIOS.

non-permanent (also temporary or interim)

A designation applied to any version of a file included in a Versions project that has not been defined as permanent. The oldest non-permanent version is automatically deleted when the maximum number of versions is exceeded.

To purge a file of its non-permanent versions, change the status of a file version from non-permanent to permanent, or change the maximum number of non-permanent versions to be stored, the file must be locked. When project security is enabled, only an administrator can perform these operations.

numeric

A value consisting of the valid digits 0-9.

omega storage

A method of file version storage in which all versions of a file are saved in their entirety. Omega storage eliminates the long processing times necessary to reconstruct complex binary files stored in delta or reverse delta format and can actually consume less disk space than other storage methods used on binary files.

Versions stores all binary files in omega format. You can also store text files in omega format. This is a time and often a space-saving decision for large, rapidly changing text files such as Windows help files (.rtf).

owner

A Versions-maintained attribute of an audit entry. An audit entry's owner is the subproject or file receiving the action. Note that an owner is NOT an individual.

permanent

A designation applied to a specific version of a file included in a Versions project. Making a version permanent prevents Versions from deleting it from the project vault when the maximum number of non-permanent versions has been reached or when the file is purged. Permanent file versions can only be deleted using **History**⇨**Delete**.

To change a version's status from permanent to non-permanent (or vice versa), the file containing that version must be locked. When project security is enabled, only an administrator can perform these actions.

project

A set of related files comprising a single product under Versions version control.

project database file

Also referred to as *the* project file. The database file, designated during the creation of a project, that stores project information. By default, it ends with an .stp extension. See also (but do not confuse with) project file.

project file

A file under version control; therefore, a file that is in a Versions project. See also (but do not confuse with) project database file.

project tree

The hierarchical display of a Versions project and its associated subprojects and files. The project tree is always displayed in the left pane of the project window.

project window

The window in which an open project is displayed by Versions. It contains two panes. The left pane displays the project tree and the right pane displays the files, versions, or audit log entry trees associated with the subproject or file selected from the project tree.

purge

The deletion of non-permanent versions from a file. If only one version of a file remains, that version cannot be purged. The most recently checked-in version is also never purged.

To purge a file, the file must be locked. When project security is enabled, only an administrator can purge a file.

remove from version control

To delete all information about a file or subproject from the project database file (.stp) and the vault. To remove a file, the file must be locked. When project security is enabled, only an administrator can perform either of these operations.

response

Any of a number of replies to a topic, that along with the topic, form a hierarchical structure called a topic tree.

right-click

The action of pressing and releasing the secondary mouse button. Since most mice are configured for right-handed operation, the left button (under the index finger) is the primary button and the right button (under the middle or ring finger) is the secondary button.

root subproject

The topmost folder in the project tree hierarchy.

scope buttons

The buttons at the top of the right pane. Depending on the contents of the right pane, one or all of the buttons may be disabled.

 No Descendants

Displays information only for the item selected from the project tree.

 Immediate Descendants

Displays information for the selected item and its children (when the selected item is a subproject).

 All Descendants

Displays information for the entire branch. This includes information about the selected item, its subprojects, and their files.

secondary sort

Sorting items by group in a list that is already sorted (primary sort). For example a file list might be sorted by extension, then (in a secondary sort), sorted by name within groups of the same extension.

Click a column header to perform a primary sort; right-click a second column to perform a secondary sort.

server

A computer or system that provides services to clients. The clients may be other computers.

sort or primary sort

To place items in ascending or descending order in the right pane of the project window based on the value in the column whose label you click. Depending on the values in the column, the values are sorted numerically or alphanumerically. Click once to sort. Click a second time to change the sort order from ascending to descending or vice versa. Right-click to create a secondary sort.

subproject

A set of related files comprising a discreet portion of a product under Versions version control.

team member

Any of the project users. Sometimes used to denote a user that does not have administrator status.

team list

The list of team members for the project. Select the Team tab to display the list and information about each team member.

test command

A command that you specify so that Versions can perform, in proper sequence, an automated test procedure for a project or subproject's product.

text file or ASCII file

A file that contains only printable text characters, spaces, carriage returns, and sometimes tabs and an end-of-file marker, without any formatting codes.

time stamp

Information maintained by Versions about files, versions, and audit entries.

For file versions: The date and time that the file was checked into Versions.

For files: The date and time for the working file.

For audit entries: The date and time that the event occurred.

unlock

The process of releasing a locked file so that others can check it in.

user

An individual identified to Versions and given access to one or more projects and their component subprojects and files. When project security is enabled, that access is protected by password. A user is also referred to as a team member.

vault

A directory where versions of the files that are under version control are stored. This directory´s name ends with a .vlt extension and is a subdirectory of the directory where the project database file (.stp) is stored.

vault file

A file in the project vault that stores versions of a project file.

version or revision

A copy of a file stored in the project vault. One or more of these versions may exist.

version attributes

Data stored for each version of a file under version control, including version number, time stamp, author, file size, permanent or non-permanent status, version labels (if any), and reason for check-in.

version control

Version control is the process of storing and tracking the various changes (versions or revisions) to one or more files. A version control system maintains the version history generated as the files evolve into their final forms.

The main advantage of using an automated version control system is fast, easy recall of previous versions. If you are on a network, Versions also provides a quick, easy-to-use method of obtaining versions of files being generated and maintained by other users working on a project with you.

version control system — Application software to help manage multiple versions of the same file.

version label — A version label provides a convenient method of identifying a version of a file by name. This is valuable when you are checking in a group of files that may need to be checked out together. When multiple files are involved, each file will generally have a different number of versions. By selecting a version label, you can check out all files related to a specific change by referencing that version label. Every version within a build or milestone has the same version label.

version number — The version number is an identification number assigned by Versions when a version is checked in. You can use this number to access the version. Versions uses it to determine if you need to check out versions of the files currently stored in your working directory.

Visual Diff — A companion program to Versions. Visual Diff compares two text files or two versions of one text file and shows the differences (if any) between them.

working directory — The directory designated to hold the working copies of files that have been checked out of the Versions project vault. It is also the directory from which files are added to or checked in to a subproject.

working file — A copy of a file under version control which has been checked out of the vault so it can be reviewed or edited. Versions copies working files to the designated working directory on a workstation. This is in contrast to the stored file versions that are located in the vault.

workstation database file

The file stwork20a.db is the workstation database. It can be stored in the directory where Version 2.0 is installed, but it is usually in the Windows directory. The database contains information about what versions of files are in your working directories and so forth. For example, Versions uses it to determine what needs to be checked in or out.

workstation options

User-selectable choices for the behavior of Versions on a specific workstation.

Appendix B
Your Computer Clock

Versions does not rely exclusively on your computer's clock and the date and time stamp of your working files to determine what needs to be checked in or checked out. Versions will still function properly, even if your computer's clock is not in sync with other team member's clocks.

However, most compilers and development environments are still sensitive to the date and time stamps. Files that were checked in by another member of your team will, by default, have the date and time stamp set by your teammate's clock, even when you check the file out. If your clock is not in sync with your teammate's, you may experience problems compiling or building the file. Therefore, we still recommend using whatever service your network server supports for synchronizing all workstation clocks.

For Windows networks, use:

```
net time \\server_name /set /yes
```

Appendix C
Using Versions 2.0a and StarTeam Together

Since Versions and StarTeam 2.1 both use the same format and file extensions (.stp and .vlt), you can share and view files between applications. In fact, Versions and StarTeam 2.1 can access the same projects simultaneously. However, StarTeam offers additional functionality. It allows you to add or edit defects that you can only view in Versions. StarTeam Workstation 2.1 also allows you to add threaded conversations to projects, subprojects, and files.

Opening a Project

When you open a project with Versions, you must know the name of its project database file (.stp) and have read/write access to the location where it is stored. This gives you file server access (also called local access) to the project. With file server access, data is processed on your workstation.

When you open a project with StarTeam Workstation Professional 2.1, you can open it for file server access or for client/server access. With client/server access (also called remote access), data is processed on the server and only the results are sent to your workstation. Client/server access is possible only when StarTeam Workstation Professional is used with its optional companion product, StarTeam Server. (StarTeam Workstation only provides file server access.)

The following procedure assumes that you are using file server access for both Versions and StarTeam Workstation 2.1.

To open a project:

1. Open either Versions or StarTeam Workstation 2.1.
2. Select **Project** ⇨**Open...** from the menu bar.

   The Open Project dialog opens.
3. Use the Look In drop-down list box (and the list box beneath it) to select the drive and directory where the project is located.
4. Double-click the project database file (.stp) to be opened.

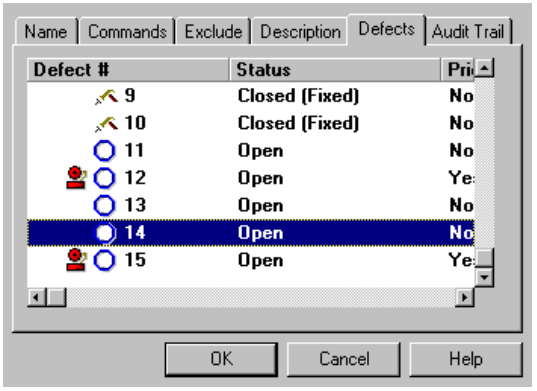
   The Log On dialog opens.
5. Type your password if one is required.

   A project window displays the project and its subprojects.

Viewing Defects in Versions

The Defects tab contains defects associated with the selection from the project tree. Although Versions does not have full defect tracking capabilities like StarTeam Workstation, limited access is provided.

Two icons might precede each defect. One indicates its status; the other appears only if the defect is a priority.



To view defects created in StarTeam Workstation associated with a subproject:

1. Click the subproject, then select **Project** ⇨ **Properties...** from the menu bar.

   Or,

   Right-click the subproject, then click **Properties...** on the context menu.

   The Project Properties or Subproject Properties dialog opens.

2. Select the Defects tab.

   The Defects are displayed in the list box.

Use the right and left scroll arrows to view all the columns in the defects list:

Defect # — Number that StarTeam Workstation assigned to the defect when it was first submitted.

Status — Indication of the defect's progress from open to resolved to verified to closed. This is referred to as a defect's life cycle.

Priority — Indication that the defect is a:

- Priority
- Not a Priority

Severity Seriousness of defect:

 Severity High (red light)

 Severity Medium (yellow light)

 Severity Low (green light)

Responsibility Person who is responsible for the defect.

Synopsis Short explanation of the defect.

Date Entered The date the defect was submitted.

Build Tested Build in which the defect occurs.

Build Addressed Build in which the defect is resolved.

Author Person who submitted the defect.

Type Type of defect:

 Defect

 Suggestion

Owner Subproject or file to which the defect is attached.

Appendix D
Upgrading to StarTeam 2.1

Versions can be easily upgraded to the StarTeam 2.1 product family which offers additional functionality such as:

- Client/server architecture (StarTeam Workstation Professional)
- Defect tracking
- Threaded conversations
- Internet, WAN, and dial-up connectivity.

Versions lets you get started with an effective version control solution. StarTeam offers a more advanced and robust software configuration management solution for your growing projects. All of your Versions projects can be accessed in StarTeam Workstation.

StarTeam 2.1 includes StarTeam Workstation Professional 2.1, StarTeam Workstation 2.1, StarTeam Server 2.1, and StarTeam Web Connect 2.1.

StarTeam Workstation Professional

StarTeam Workstation Professional is a tightly integrated suite of SCMtools including version control, visual differencing, defect tracking, threaded conversations, build management, audit logs, status reports, and an advanced project repository. StarTeam Workstation Professional can be used for file server access or for client/server access and focuses on the larger team issues that affect delivery of applications.

StarTeam Workstation StarTeam Workstation offers the same functionality as StarTeam Workstation Professional with the exception of remote access. StarTeam Workstation is ideal for small groups using local area networks.

StarTeam Server StarTeam Server provides support for distributed StarTeam Workstation development teams and mobile team members by providing access to projects through Internet, intranet, WAN, or dial-up connectivity. This removes any limitations on where development team members or consultants are located. In addition, StarTeam Server uses client/server technology to increase performance and reduce network traffic. It provides a scaleable, secure solution that serves the needs of even the largest development team.

StarTeam Web Connect StarTeam Web Connect provides access to application and web site projects via a web browser. This revolutionary way of working with and managing projects allows a wide range of users to participate in an easy, secure, and cost effective development process. StarTeam Web Connect allows different views of the same project and dynamically gives each user the appropriate project view based on the rights assigned to that user by the administrator.

As part of an application development project, StarTeam Web Connect provides a way for beta users, testers, consultants, and others to interact, provide feedback, and review information.

As part of a web site project, it allows the webmaster to create a framework that meets the needs of everyone involved with content, graphics, testing, and responding to user input. In addition, StarTeam Web Connect is ideal for controlling web site configuration and can operate standalone or in conjunction with a commercial HTTP server.

Who needs StarTeam 2.1?

StarTeam 2.1 offers an Integrated Team Environment (ITE) by using one interface for all resources. StarTeam 2.1 runs on Windows 95 and Windows NT, and supports all popular file-based programming environments including C++, Visual Basic, Delphi, JAVA, and Perl. Developers work more productively as a team because they have quick access to all project information and each other across the LAN. With the optional StarTeam Server 2.1 and StarTeam Web Connect 2.1, developers and others can also access projects over Internet, intranet, WAN, remote dial-up, and web browser. StarTeam 2.1 is recommended for:

- Projects requiring integrated version control, defect tracking, and threaded conversation
- Web Site management
- Managers and developers needing to know project status
- Local and distributed development teams
- Interdepartmental teams including beta users, testers, writers, and consultants
- Projects requiring strong security

Feature Comparison

Feature	Versions	StarTeam 2.1
SCM Functionality	Version control	Version control
	Visual differencing	Visual differencing
	Build management	Build management
	Audit logs	Audit logs
	Project repository	Project repository
	File, version history, team, and audit log charts & reports	File, version history, team, audit log, charts & reports
		Defect Tracking
		Threaded Conversations
Scalability	FAT Client/LAN	FAT Client/LAN
		WAN/Internet
Communication	N/A	Internet
		Dialup
		WAN
Web Browser Access	N/A	Yes
Web Site Management	N/A	Yes
Security	Limited	Advanced

To order StarTeam Workstation Professional 2.1, StarTeam Workstation 2.1, StarTeam Server 2.1, and/or StarTeam Web Connect 2.1, please contact StarBase Corporation at 1-888-STAR-700 between 9 A.M. and 5 P.M. Pacific time, Monday through Friday.

Appendix E
StarBase Technical Support

If you have a question about Versions, you can:

1. Consult the online help and written documentation.
2. See the README.WRI file that comes with Versions.
3. Post a message to our CompuServe forum.

   `GO STARBASE`

   Then visit the StarTeam message and library sections.
4. Send a message to support@starbasecorp.com.
5. Access our web site at:

   `HTTP://www.starbasecorp.com`

   Check for new support questions and answers.

If you can't find the answer from any of these sources, or if you require information about product registration or replacement disks, call StarBase Technical Support Services at (714) 442-4460 between 9 A.M. and 5 P.M. Pacific time, Monday through Friday.

When you call StarBase Corporation Technical Support Services, please have the following:

- The version number of the Versions software you are using.
  (From Versions, select **Help** ⇨ **About Versions...**.)
- The exact error message you received, if any.
- A precise explanation of what you were doing when the error occurred.
- The version of Windows you are using.
  (Select **Start** ⇨ **Settings** ⇨ **Control Panel**. Then select the System icon. The version appears on the General property page.)
- The type of hardware you are using.
- The network operating system you are using, if any.

StarBase support services are subject to StarBase prices, terms, and conditions in place at the time the service is used.

For information about product upgrades and pricing, call StarBase Corporation at 1-888-STAR700 or (714) 442-4500 between 9 A.M. and 5 P.M. Pacific time.

Appendix F
Questions and Answers

This appendix answers commonly asked questions about Versions.

Q: How can I improve Versions' performance?

A: Do some of the following:

- Organize your subprojects for efficient access. It is best to make the project hierarchy as flat as possible. Wherever you can, make subprojects siblings rather than parent and child.
- Pack the project database file (.stp) using the Project Maintenance utility.
- Use vault file compression.

  (Select **Project** ⇨ **Properties…**, then the Default tab. Select the Use Compression check box.)
- Disable the audit log.

  (Select **Project** ⇨ **Properties…**, then the Name tab. Clear the Audit Log check box.)

Q: How does Versions know what files are ready to check in or out?

A: Versions uses a workstation database (stwork20a.db) and the time stamp on files to determine what you should check in or out.

Q: What if the workstation database file (stwork20a.db) is corrupted or lost?

A: If you lose the workstation database file (stwork20a.db), Versions recreates it automatically. To resynchronize your workstation information, check out your project files again.

Q: Do I need to do backups?

A: Versions project vaults are not a replacement for backups. In fact, it is critical that you back up the following:

- Project database files (.stp)
- Their vault directories (.vlt)
- The files within the .vlt directories

In certain environments, open files are not backed up. Because a wide audience can access a Versions database project file at various times of day, a project may always be open by at least one user.

To make sure that project database files (.stp) are backed up even if they are open, copy them and their associated vault directories (.vlt) and files to another server location and have that location backed up instead. Perform this copying process before the backup process or automate it using the operating system's scheduling service.

Q: How is your project information stored? Is it stored in a known database format or do you store it in a proprietary format?

A: We are using the Access database format to store much of the data. Actual file versions are stored in StarBase's own proprietary format.

Q: Is your Versions product integrated with the Visual Basic, Visual C++, or Delphi environments?

A: Versions supports any file-based development environment. You can launch your development environment from within Versions as from File Manager or Windows 95 Explorer.

For other IDEs, such as Microsoft's Developer's Studio, Versions' command-line utility allows you to perform version control functions if the development environment allows you to add tools to menus.

Q: How does your product deal with shared files? Most of our projects have a set of files in common.

A: Since Versions permits you to have more than one project, you can put the shared files in a single project (named, for example, Shared Files). When you open a project that uses the files in the Shared Files project, you can open the Shared Files project as well. You perform actions, such as check-ins and check-outs, in two separate projects.

The alternative is to include all your "projects" in one Versions project. Each of your projects becomes a subproject of the one large project. Shared Files would be one of those subprojects.

Q: I am currently unable to check in changes I have made to a file. What is the problem?

A: You need to lock the file. A number of different operations require that the file be locked. For a complete list of these operations, see the "Locking and Unlocking Files" section in Chapter 4.

Q: Can I bypass the Log On dialog when security is disabled?

A: You cannot bypass the Log On dialog. Versions needs to know your user name to note author information and monitor locker data. The logon name you use for one session becomes the default logon name for the next session. This allows you to log on to a project that does not enforce security quickly by pressing Enter or clicking OK.

Q: When I try to check in changes that I have made to one of my files I get this message: "File in use by another, try again later." I know no one else is using the file. Why do I get this error message. What can I do so that I can check this file in?

A: This error message occurs as a result of an abnormal interruption in the check-in or check-out process. As a part of these processes, Versions creates temporary files in the project vault directory (.vlt directory).

When the process completes successfully, Versions automatically deletes these files. When the process is interrupted, these files need to be deleted manually.

Use the File Manager or Windows 95 Explorer to look for files in the project vault directory with unusual names. Vault files usually have names like vv1.vvf or vv10.vvf.

The abnormal file names are similar to __cvn___. or have an underscore (_) in the last position of the file extension. They are generally 0 bytes in length. Delete these temporary files. Sometimes you have to exit Versions and restart Windows before the error is resolved. Be extremely careful not to delete any files with the .vvf extension.

Index

W