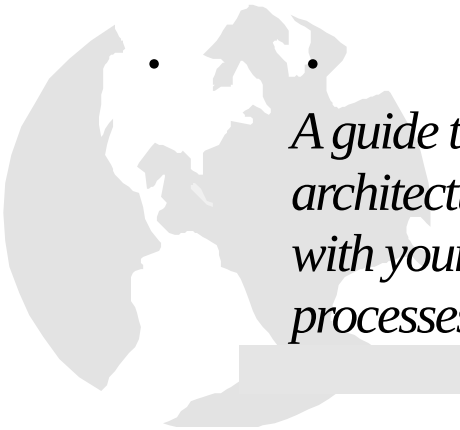
StarTeam Best Practices



A guide to StarTeam's general architecture and how to use StarTeam with your software development processes

Table of Contents

Introduction

StarTeam has revolutionized software configuration management (SCM) applications by saving your company time, money, and code integrity. Since StarTeam's inception, we have learned from our own experiences and that of our customers about the best ways to install, configure and use StarTeam. We'd like to share this knowledge and information with you. This document includes:

- **StarTeam Overview**

  A general description and introduction to the StarTeam product line.

- **The StarTeam Model**
  An overview of the object-model used in StarTeam which reviews the differences between *project-oriented* systems like StarTeam and *file-oriented* systems like PVCS, Visual SourceSafe, and MKS.

- **Software Development Processes**
  A review of some of the basic processes used in software development and how StarTeam supports these processes.

- **Capability Maturity Model**
  Some techniques for those organizations that strive for compliance with the Software Engineering Institute's Capability Maturity Model.

- **Use Cases**
  A number of technique examples from our users and technical support team.

- **Tips**
  A number of usage tips from our users and technical support team.

We will continuously update this document with the latest information. Please visit our web site, www.starbase.com, for the latest version of this document. We also welcome you to send us your real-life examples, experiences, comments and suggestions that will allow us to improve this document and encourage collaboration. Send us an email: support@starbase.com.

StarTeam Overview

The StarTeam product line is comprised of the StarTeam VirtualTeam Server and its clients: StarTeam, StarDisk, Universal Edition, and Web Connect. When the administrator sets up the StarTeam VirtualTeam Server, s/he configures the server using the database of choice (Microsoft Access, Microsoft SQL Server, IBM DB2, Informix, Sybase SQL Server, or Oracle). Users can be located at one site or widely dispersed across the United States or across the globe. Users access StarTeam VirtualTeam Server via one of the clients: StarTeam, StarDisk, Universal Edition, or Web Connect.

StarTeam, a Windows application, provides an intuitive GUI that displays the project, views, folders, files, etc. The tabbed panes make navigation and deployment easy whether you're working on bug fixes (change requests), product discussions (topics), work assignments (tasks) or developing code (files). StarTeam integrates with many popular IDEs, for example, Microsoft's Developer Studio, PowerBuilder, Delphi, and Oracle. It interoperates with PVCS and SourceSafe allowing you to convert existing SCM projects to StarTeam. It also offers a command-line interface.

StarDisk allows users to access file revisions over TCP/IP networks through a virtual StarDisk drive. StarDisk's integration with Windows Explorer provides transparent access to some of the benefits of StarTeam.

The Universal Edition makes the command-line interface available on platforms that support Java version 1.1 or higher. This allows UNIX users to access StarTeam.

Web Connect users access project repositories via a standard web browser. Web Connect allows users to check files in and out of a StarTeam, PVCS or VSS repository, as well as, create, edit and report on change requests and participate in team discussions.

Customized clients can also be created using StarTeam's SDK.

StarTeam is ideal for all types of businesses. Some of our diversified customers include Anderson Consulting, Lockheed Martin, and Frito-Lay.

The StarTeam Model

To make the best use of StarTeam in your software development process, you need to become familiar with StarTeam terminology. These terms may seem foreign at first, but you will discover that the StarTeam model is more applicable to your business practices than the models found in earlier generations of SCM systems. Also, the flexibility of the StarTeam model allows you to finally manage *all* of your information assets, source code as well as documentation, through a single coherent system.

The StarTeam Repository

The backbone of the StarTeam system is the StarTeam repository, which is supported by the StarTeam VirtualTeam Server. This repository is an object-oriented data store that supports object versioning, linking and configurations. Any object, known as a StarTeam item, stored in the repository has its history recorded so that a prior state of the item may be retrieved and restored. StarTeam items may be linked to any other item in the repository so that the relationships between the various information assets can be maintained and used in your work processes. Configurations containing many items can be created, maintained and restored through the services of the StarTeam repository.

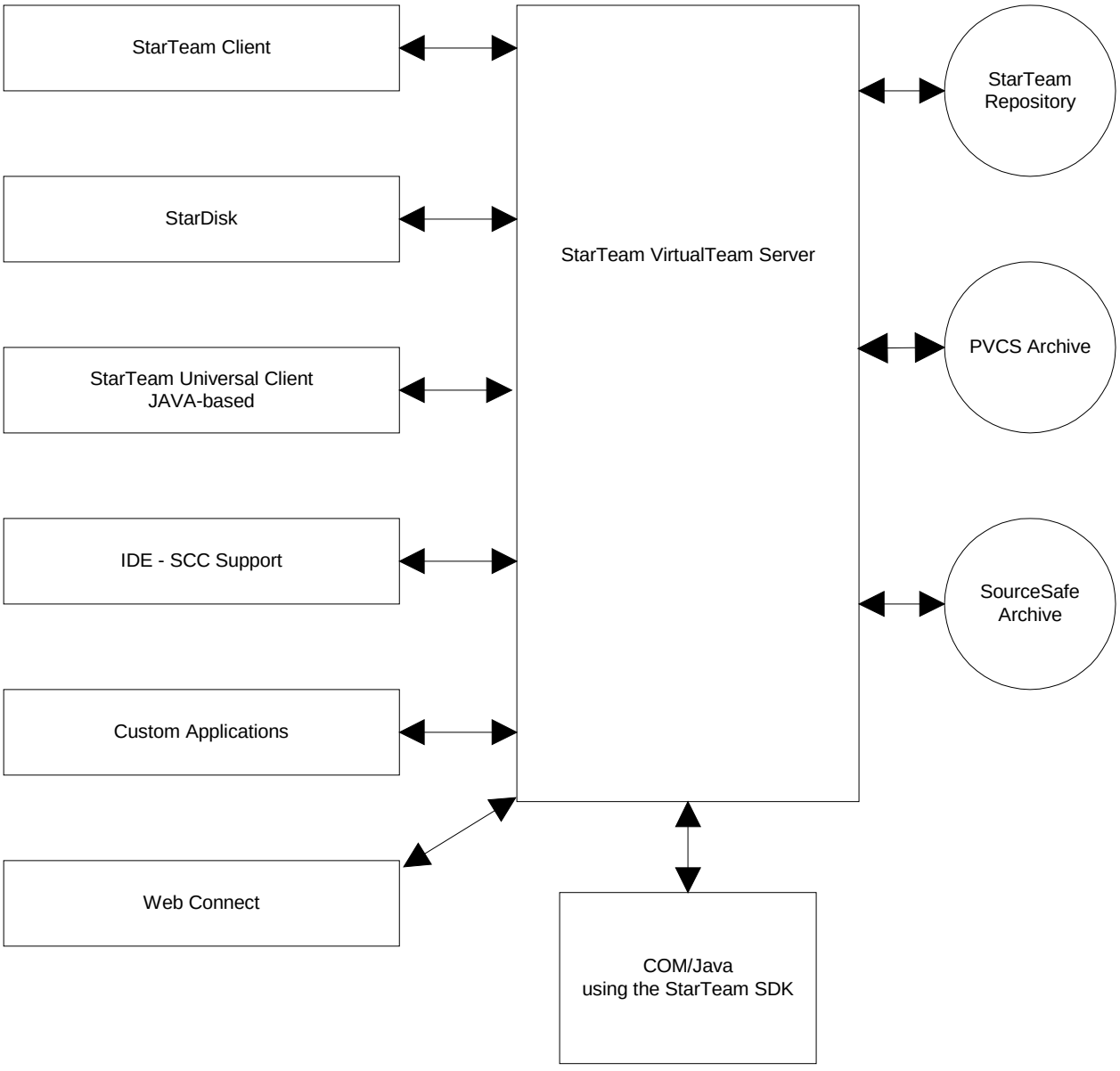
Client Server Architecture

Access to the StarTeam repository is through the StarTeam VirtualTeam Server. This means that your archive files are fully protected. Products such as PVCS, MKS and SourceSafe require that the revision archives be accessible by all users as shared files. This makes these archives, and the information assets they store, also accessible to attack by computer viruses or disgruntled employees. With StarTeam, the only process that needs to access these archives is the StarTeam VirtualTeam Server.

All StarTeam clients, whether it is the StarTeam Window GUI, command-line interface, IDE integrations, StarDisk, or a custom application using the StarTeam SDK, communicate with the StarTeam VirtualTeam Server using TCP/IP. StarTeam, the Windows application, can also use NetBEUI, IPX/SPX or Named Pipes. Since StarTeam has been optimized for the Internet, remote users can access the StarTeam repository knowing that all data transfer is compressed and encrypted.

A final consideration when reviewing the StarTeam Client Server Architecture is that StarTeam lets *you* decide on the database to use. You can select from Microsoft Access, Oracle, Microsoft SQL Server, Sybase SQL Server, Informix and IBM DB2, all industry standard databases that should be familiar to your Database Administrator. For the first time, you can pick the database that is your corporate standard for managing your information assets.

Figure 1: StarTeam Client/Server Architecture



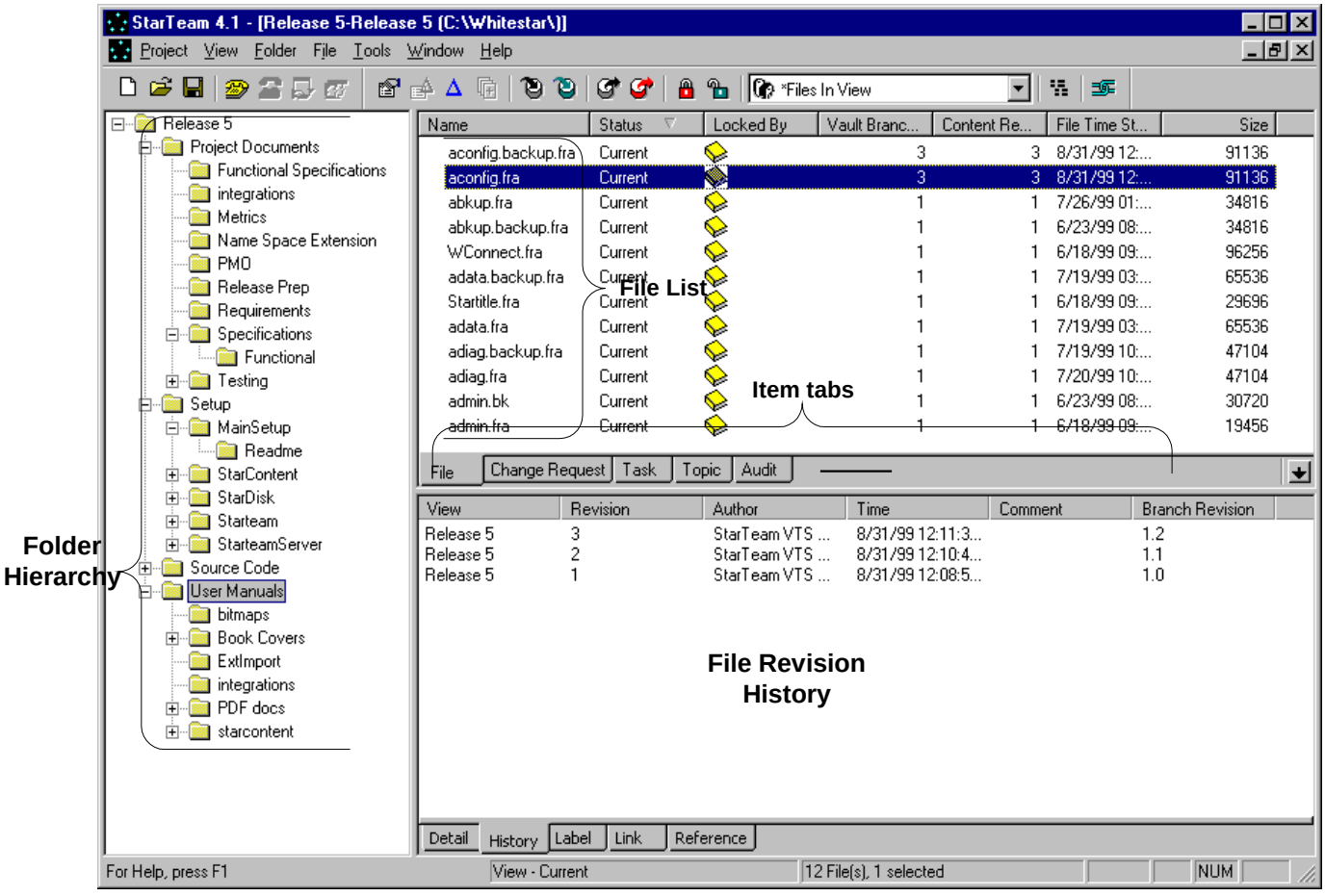
Project-Orientation

Older SCM applications such as PVCS, SourceSafe and MKS, are directed towards individual files. These are referred to as *file-oriented* version control systems. Each file added to the system has its revisions stored in a specific archive file and this one-to-one mapping is independent of the location where the file is placed when building the application. Some of these products, such as PVCS, do not keep track of which directories the files need to be checked out to. This information is needed to correctly rebuild historic configurations of the files. In addition the only information placed under

version control in these systems, as well as in some more sophisticated products such as ClearCase, CCC/Harvest and Continuus, are the revisions to the files.

StarTeam takes a *project-oriented* approach. In this approach, the source code and documentation files are seen as only one specific item type that makes up the complete project. In addition to the file-oriented version control features found in older products, StarTeam supports version control of the other items you need for a project, such as change requests, topics, tasks and the folder structure that stores these items. A project-oriented system also allows users to view these items in different ways depending on their role or the immediate work requirements of the project. The project-oriented approach is a *superset* of the features found in the products that implement the file-oriented approach

Figure 2: StarTeam, a project-oriented SCM application



to SCM.

Items

The StarTeam model uses items, such as files, change requests, topics, responses, and audit entries. The most commonly used items are versionable, that is, StarTeam stores a revision history of the item and permits you to view and compare the contents of different revisions.

Items may also be branchable; that is, the can be derived from other items that become their ancestors. Items may have several completely different revision histories with common ancestries. In the case of a text file, for example, the branched item can later be merged with the file 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.

The concept of branching is not often found in documentation management systems. However, this ability is fundamental to software configuration management. Developers often need to make minor or major changes while still preserving the original development path.

StarTeam's collaborative framework architecture supports several types of items but is designed to permit the development and addition of more items based on customer demand. The following table lists the types of items found in the current release of StarTeam.

Table 1: StarTeam Item Types

Item Type	*Versionable*	*Branchable*
File	Yes	Yes
Change Request	Yes	Yes
Task	Yes	No
Topic	Yes	No

Projects

StarTeam uses projects, views and folders to organize the items stored in the StarTeam repository. A StarTeam project is best thought of as a collection of closely related views. Each view represents a configuration of items from the repository and can support a different line of development on the same code base. Folders cluster items into groups. For example, you might want to check out all the files in a folder to work on a particular feature of your product. However, there are no restrictions on items found in different projects. The items in a repository can be moved or shared between any views, regardless of the projects the items and views are found in.

Projects provide an additional layer of organization, but they allow you provide a hierarchical structure for the views as well as assign rights at the project level. How projects are used is largely up to you.

You might create a project for each product that your company produces. Or, depending on how you build your product, you might prefer to create a project for each of a product's major components. Using a separate project for each product components provide flexibility as each component can easily be labeled, branched, and sent through its own promotion model sequence.

Views

When you open a StarTeam project, you must either accept the default (or main) view or select an alternate view. The default view for a project typically contains the configuration that is used for primary development. Additional views can be *derived from*, that is created based upon this view, and can take on different behaviors. The selected view presents the items found in a particular configuration.

Views typically have names such as Baseline, 4.0 Maintenance, Special 4.0 for Australia, and 5.0 New Development. They represent configurations of items and support different development baselines on

the same code base. Views can be compared and merged. For example, you might want to merge files from both 4.0 Maintenance and 5.0 New Development back into the Baseline view eventually.

You can create and use views that:

- **Dynamically show the source code and document changes of your project.**
  This is the typical use of the default (or main) view in a project when the Current Configuration option is specified from the View menu's Select Configuration command. This dynamic view shows all the items as they change and is used for collaborative development.

- **Reference a subset of items found in the original view.**
  These are often called *reference views*. Any change made in the new view changes the same item in the original view. This is because the subset view contains *references* to the original items in the original view and does not branch when changes are made. Typically reference view have names such as Development View or Documentation View, exposing only items of interest to, for example, developers or writers.

- **Are read-only and based upon a specific state of the original view.**
  This is typically done for convenience so that the revisions of items used in product releases can be easily located. For example, a 4.1 Release view might be used to rebuild 4.1 in the future or to allow a company that wants to purchase your source code to review that source code after signing a tentative agreement.

- **Permit branching of the items in the new view.**
  This view can be used to modify the items found in a specific view state without affecting the main development. This is typically done when creating and maintaining a maintenance baseline.

An important feature of a view is that you can reconfigure it to show the items as they existed in that view at an earlier point in time or based on a view label or associated promotion state. You *roll back* a view using the View menu's Select Configuration command. A rolled-back view is read-only, showing a precise state of the items and no longer permitting changes to them.

> *Use the View menu's* Select Configuration *command to locate the file revisions that were checked in as of a specific time as well as the state of change requests, topics, and tasks as of the roll-back time.*

Folders

Each StarTeam view contains a folder hierarchy used to organize its items. Folders reflect the logical organization of the configuration represented by the view. Folders typically have names such as Source Code, Schedules, and User Manuals. They group the items based on who needs to access them or on the closely related nature of the set of files in them. While folders can be organized into any hierarchy, the structure typically follows the structure of the working folders that the files are checked out to.

Folders can also be useful when you need to create different configurations of shared items. You can share folders, files, change requests, tasks, and topics between and within views so long as the views use the same server configuration. When a folder is shared, users of both views can access its contents, including child folders and their contents.

Sharing folders can be an important part of setting up a view. For example, suppose all products use the company's general libraries to some extent. Even though those libraries are not maintained by the developers of a given product, the product is based on some revision of the source code in them and must be compiled with it. Therefore, some of the library folders should be shared into the product's view.

You use Ctrl+Drag, to share a folder or an item from one location to another. By sharing, you create a reference to the original folder or item. Unless the shared folder or item's behavior is set to branch on change, all changes to it are changes to the original.

The shared folder or item's configuration (floating, based on a label, a promotion state, or a point in time) is initially identical in both views. However, it can be modified in either. This means that the shared items can be very different in each view. Make sure you understand sharing thoroughly before you do this.

The shared folder or item loses any labels it had in the previous view. Labels cannot be moved from view to view.

View Labels

Another feature of a StarTeam view is view labels. View labels are used to identify static configurations containing specific revisions of items in a view. When you create a view label, it stores a time stamp for the view. A view label gives you a static *snapshot* of the dynamic view that it was created in.

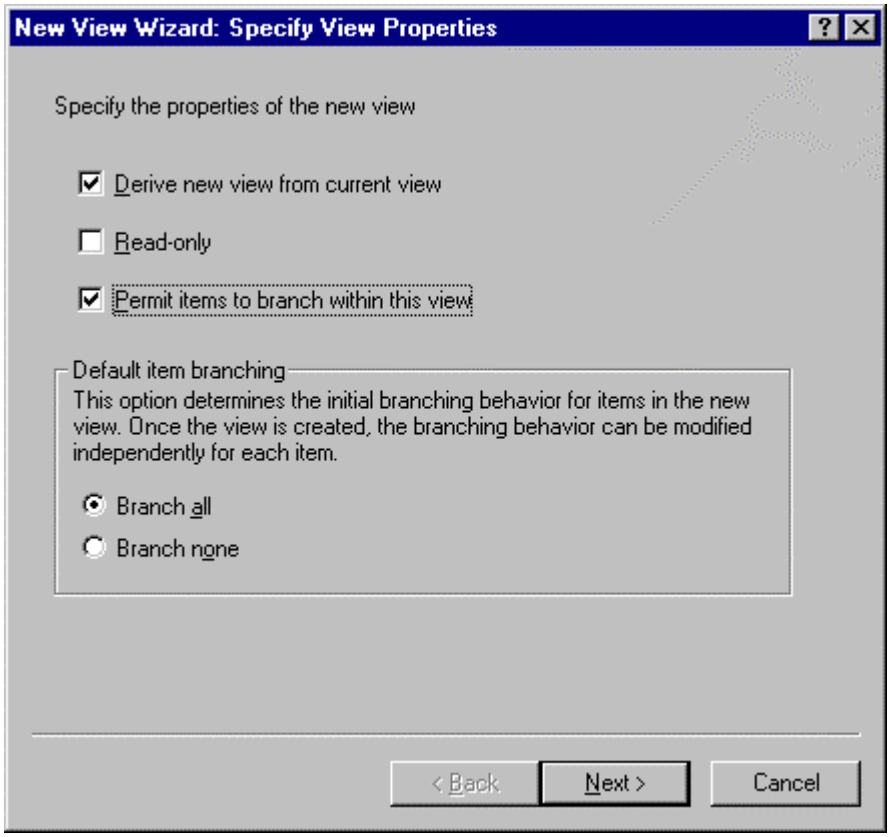
Changes can be made to the specific item revisions associated with a view label by dragging the label from one revision of an item to another in the Label pane. Typically a view label contains a small number of these label changes while the majority of the item revisions are identified by its time stamp.

> *Use view labels to indicate development milestones such as a daily build. This allows you to return to the precise configuration of specific revisions at a later time by using the View menu's* Select Configuration *command or the /CFGL option (configures the view using the specified label) from the command line.*

Branching Views

A common operation in software development is to create a new configuration that holds branched items based on some prior static configuration. This is done whenever a user wishes to perform maintenance on a previous build of the system but does not want to affect the current development. StarTeam supports this through branched views.

Create a branched view by selecting New… from the View menu and then selecting the Permit Items To Branch Within This View check box. Selecting this option allows the new view to have a distinct and independent view label namespace and the items found in the new view will be able to branch. You can elect to have every item branch when it first changes or you can defer this decision and selectively branch items in the new view.

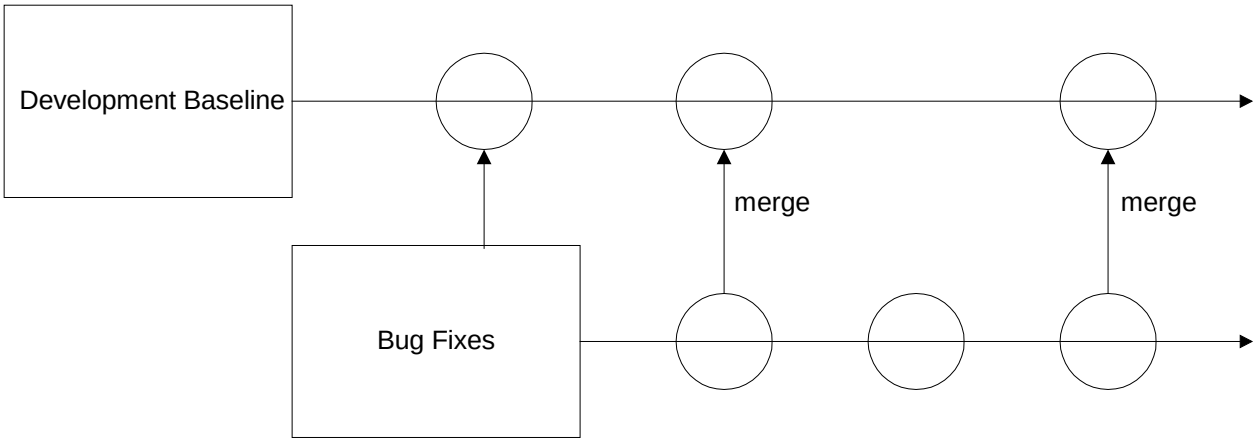


> *StarTeam's project-oriented approach allows you to determine the branching behavior for an entire configuration through the* Branch on Change *option. Unlike file-oriented systems where you need to indicate that you want to branch on each individual file, StarTeam allows you to indicate that branching should occur for a particular view based on the intended use or work process you plan for the view.*

Users unfamiliar with the StarTeam model are often confused by the fact that the view labels from the old view are not also found in the new view. This is typically due to their familiarity with file-oriented systems and *revision labels*. In these systems revision labels are unique within all branches of a particular file archive. In a project-oriented system like StarTeam, each configuration space, represented by a view that permits branching, must also have a unique view label namespace. This is because the view branches when you create a new view that permits branching. In addition, each view presents only the history of the branch of the item referenced by the view and not the item's entire history through various branches. This makes the new view an independent configuration of items and, for these reasons, the view labels found in the original view do not exist in the new view.

> *You do not have to create a new view every time you wish to branch an item. By sharing (*Ctrl+Drag*) an item from one folder to another and then setting the Behavior option to* Branch On Change*, you create a branch of an item within the same view. This gives you the same file-based branching capabilities found in older version control systems such as SourceSafe.*

Figure 3: Create New View based on Original Baseline



Merging Views

Often you will want to merge these independent branched items back into the original view or into another view. You can merge items from different views or even the same view but from different folders. The StarTeam View Compare/Merge utility performs a complete comparison of the folders, files, change requests, tasks and topics.

> *The ability to merge views allows you to make changes to items in a maintenance view and then later merge these into the main development view. Since change requests also branch, you can indicate that a change request is* FIXED *in the maintenance view while still remaining* OPEN *in the development view. Change requests can also be merged so that important information discovered in resolving the request in the maintenance view is not lost when the change request is merged.*

Linking

The ability to link any item to any other item is a powerful StarTeam feature. Many customers use this feature to record the relationships between items such as requirement documents (stored in files), specific change descriptions (stored in change requests), design discussions (stored in topics) and source code changes (stored in files). Since links can also be *pinned* (or attached) to specific revisions of linked items, you have an environment that provides complete tracability.

> *Lockheed Martin's Orlando facility passed the external Capability Maturity Model (CMM) Level 3 Audit because of StarTeam's ability to link items. As of this writing (Sept., 1999), they plan to pass the CMM Level 4 Audit using StarTeam's task component.*

File Status

One of the more unique features of StarTeam is the "heads-up" display of file information. The File Status field provides information on the relationship between the files stored in the StarTeam repository and the files stored on your workstation. Understanding these status values and how to use this information can greatly increase your productivity. The File Status values are listed in the following table.

Table 2: File Status Descriptions

File Status	*Description*
Current	Workstation file is the same as the tip revision of the corresponding file in the view.
Out of Date	Workstation file is the same as an older revision of the corresponding file in the view.
Modified	Workstation file has been modified since being checked-out from the view, but no newer revision is found in this view for this file.
Merge	Workstation file has been modified since being checked-out from the view and there is a newer revision in this view for this file.
Missing	There is no workstation file found for the file in this view.
Not in View	There is no file found in this view for the workstation file.
Unknown	There is no record of this file being checked out from this view, but a file with the same name mapping to the same working folder exists in the view. Use the Update Status command to allow StarTeam to match this file to a revision of the file in the view and provide an accurate status.

When you update the status of a file, StarTeam compares the working file with the revision you checked out and the tip (most recent) revision. For example, the file list may say that the file is Current, but someone else has just checked in a copy of it, so your status really is Out Of Date.

Updating file statuses is not the same as updating files. For example, if a file is not in your working folder, updating the status will let you know that the file's status is Missing. It will not check out the file for you so that it is no longer missing. After all, you may not want the file on your hard drive. Normally, you use a file's status to determine whether the file should be checked in, checked out, added, or ignored. Once you become familiar with the File Status field, you can use it to:

- Check in a file if its status is Out Of Date, Missing, or Merge.
- Check out a file if its status is Modified or Merge.

- Add a file to StarTeam if its status is Not In View.
- Know ahead of time that you need to merge workstation files during check-out.
- Run Visual Diff to compare a working file that is Out Of Date with the tip revision. This lets you review the changes other team members have made before checking the tip revision out.
- Check out all files from an older build by rolling back to a specific view label. (From the View menu, select Select Configuration…, then return to the current configuration to review every change made since that build was created by comparing the checked out files with their tip revisions.)
- Locate small changes that cause big problems by incrementally rolling back the view and looking for files with the status Modified. Use the History tab to determine when files were changed.

Use Cases

StarBase realizes that each company's work environment has unique structures, policies, processes and software development lifecycles. Our goal is to model your SCM methodology within StarTeam. The following is a brief introduction to some Use Cases that our current customers use and the best way to apply these using StarTeam.

I. Release Development and Main Development

You do not have to freeze the baseline during release activities with StarTeam. StarTeam handles release development and main development in a number of ways. The following brief scenarios may be familiar.

Sample Scenario 1

During the development cycle both Joyce and Mike have code changes that need to be checked into StarTeam for Release 1. Joyce finishes her work for Release 1 a couple of weeks before Mike. While Mike is still working on his changes, Joyce needs to continue working on her files for the next release without affecting Release 1. How do you ensure that the items Joyce checks into StarTeam will not stop continued development in the main baseline? And, when Mike is done with his changes, how will they be incorporated in Release 1?

Using a New View to Resolve Scenario 1

One way to handle this scenario is to create a new, but perhaps temporary, view into which Joyce checks her additional changes. Later these changes can be merged back into the baseline, if and when appropriate.

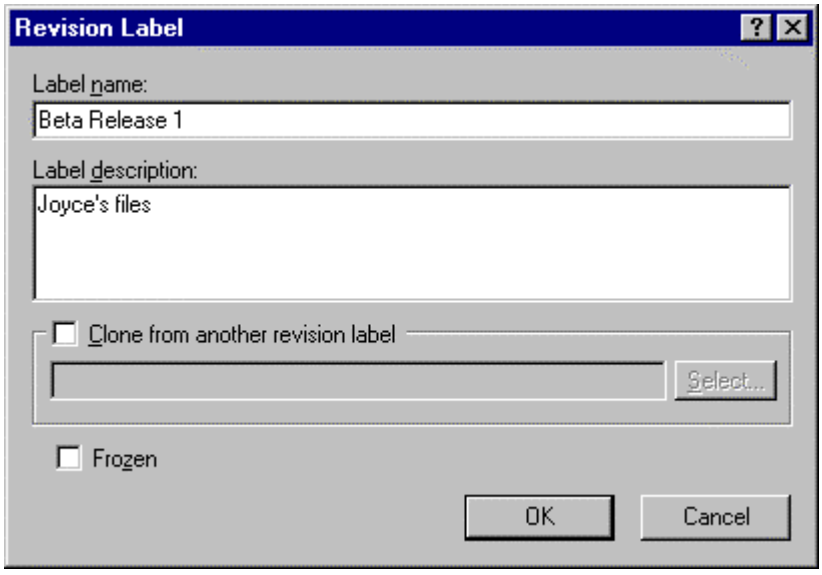
Using Labels to Resolve Scenario 1

Another way to handle this scenario is by using of revision labels and view labels. A **revision** label provides a convenient way to identify a revision or a small set of revisions using the revision label's name. This is primarily used for files.

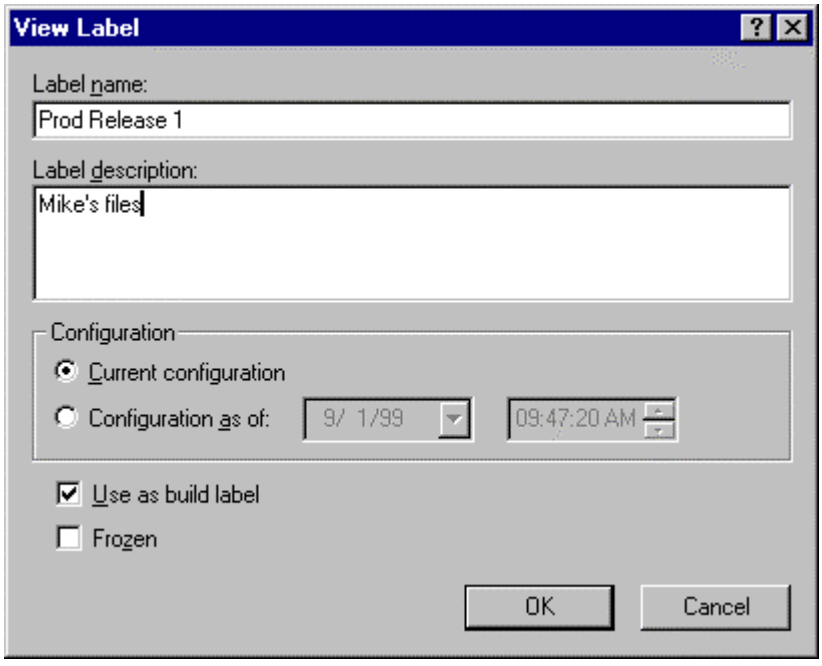
A **view** label applies to the latest revisions of all the items in a view. For example, when the project reaches a particular milestone (such as beta), you might create a view label. Then you can configure the view to return to the way it was at the time the label was applied, check out revisions as a group using that label, create a new view based on the label, or assign the label to a promotion state.

In this scenario, Joyce and Mike would perform the following steps:

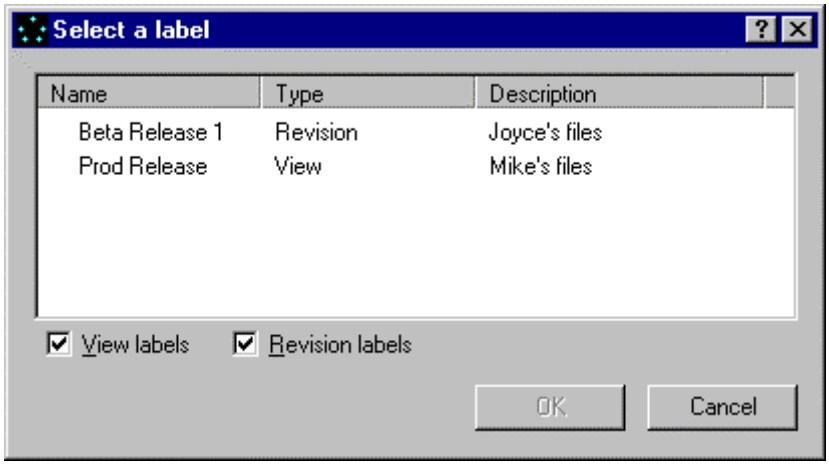
1. After Joyce is done with her changes for Release 1, she creates a revision label for her files (called Beta Release 1) and continues developing for the next release.



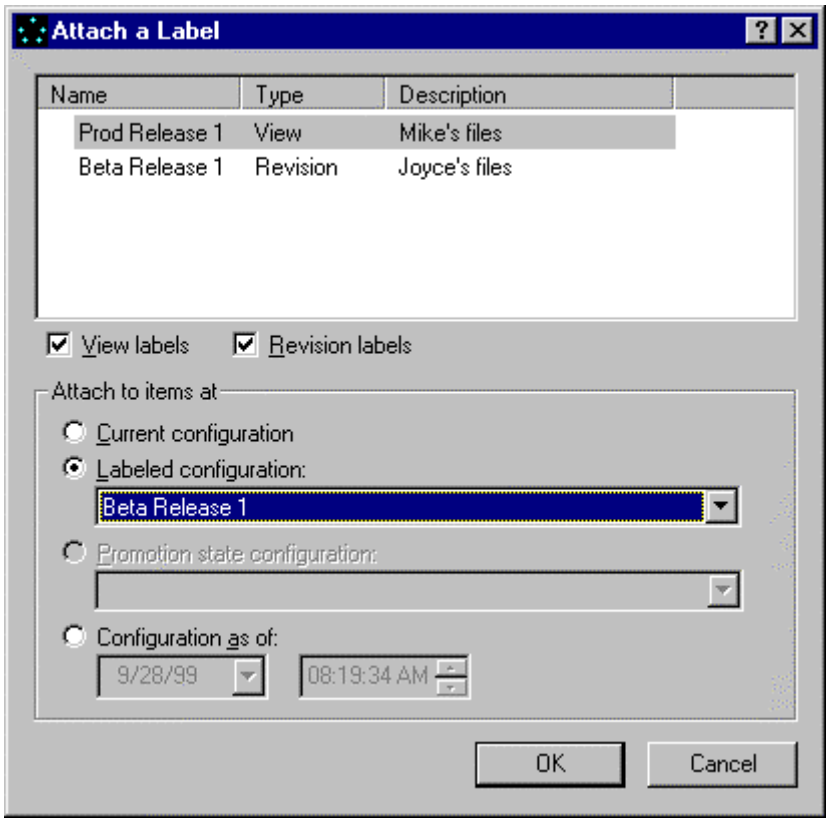
2. When Mike completes his changes for Release 1, he creates a view label (called Prod Release 1).



3. Mike then (using StarTeam 4.1) selects File >Select>By Label… to find Joyce's revision label (called Beta Release 1).



4. Once the queried files are selected (Beta Release 1), Mike moves the view label (Prod Release 1) from the tip revision of each of these files to the revision that has the label Beta Release 1 using the Attach a Label dialog.



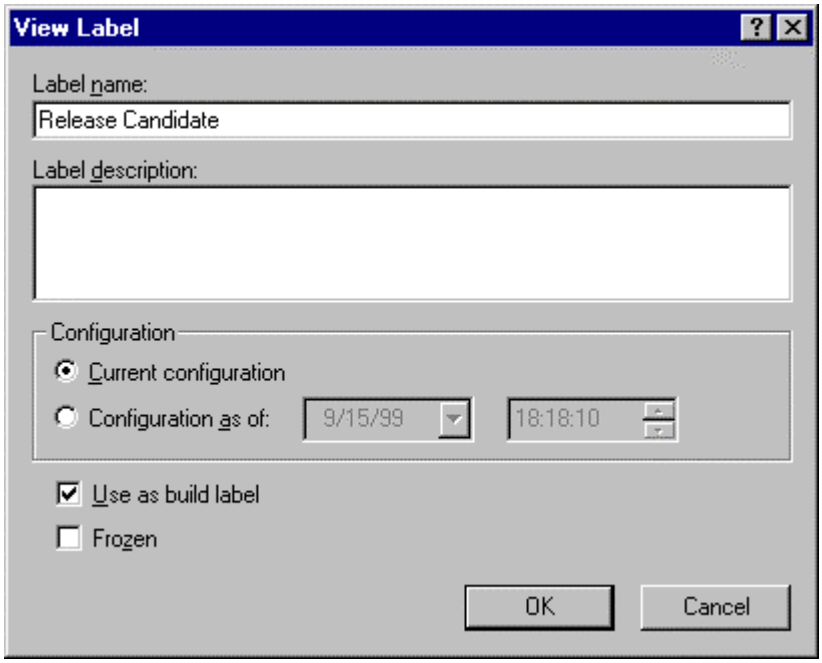
Sample Scenario 2

Mike is the project coordinator and StarTeam administrator for a group that is simultaneously releasing software to production and developing new releases. How does Mike handle the project releases while allowing other development to continue on the main baseline?

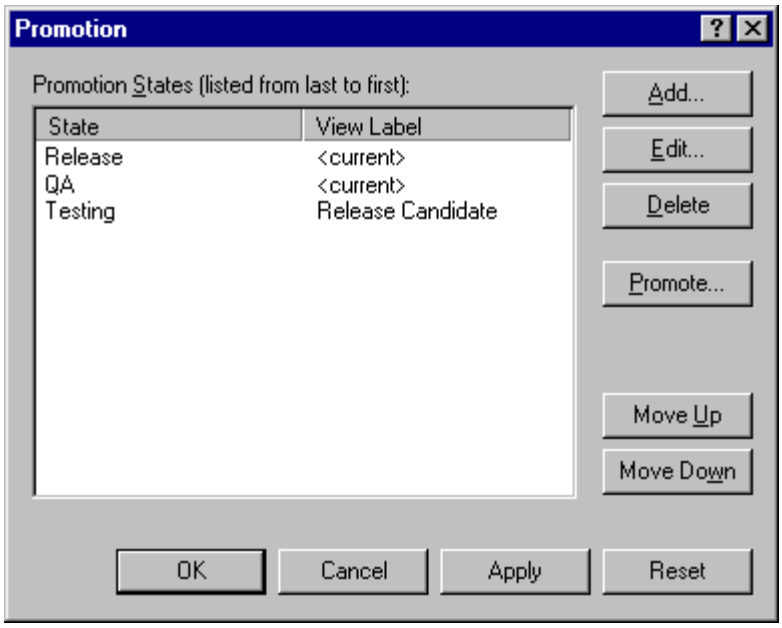
Using Promotion States to Resolve Scenario 2

StarTeam uses labels, promotion states and branched views to allow for continuous baseline development during release engineering activities. The steps in this process are:

1. Create a view label when development is complete. For example, you might name the view label Release Candidate.

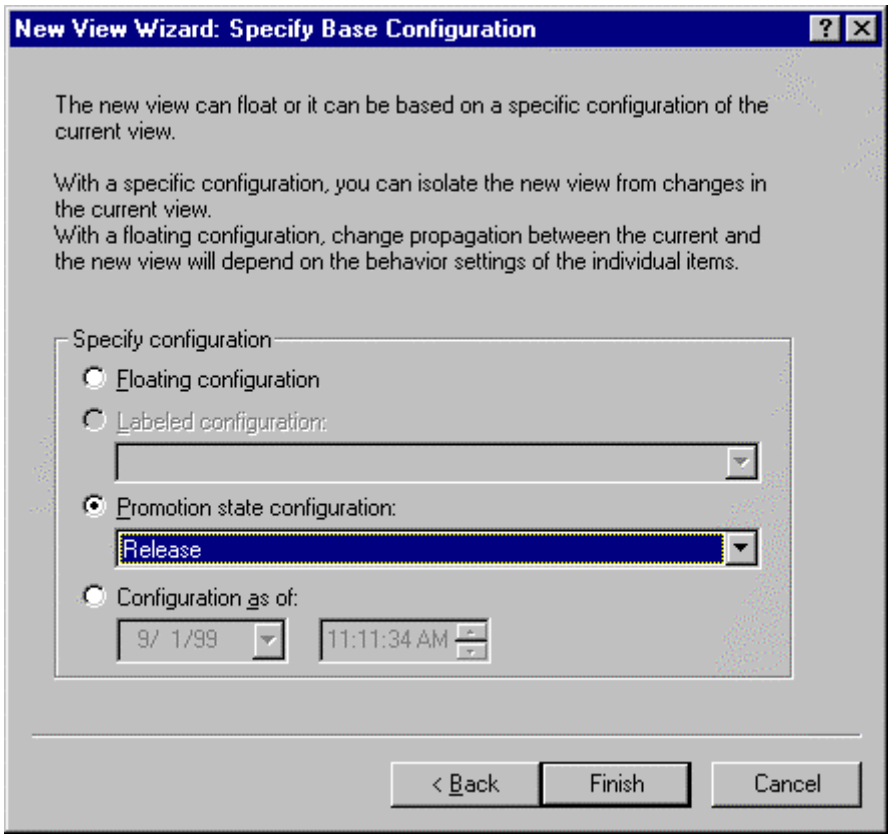


2. Select Promotion…from the View menu. From the Promotion dialog, create various promotion states (for example, Testing, QA, and Release).



3. Select Promotion…from the View menu to promote the Release Candidate view label through the various states until it reaches the final state, the Release state.

4. Create a branched view derived from the current view and assign this configuration to a promotion state. In this case, the desired promotion state is Release.



A separate baseline is created for release integration and engineering while development continues in the main baseline.

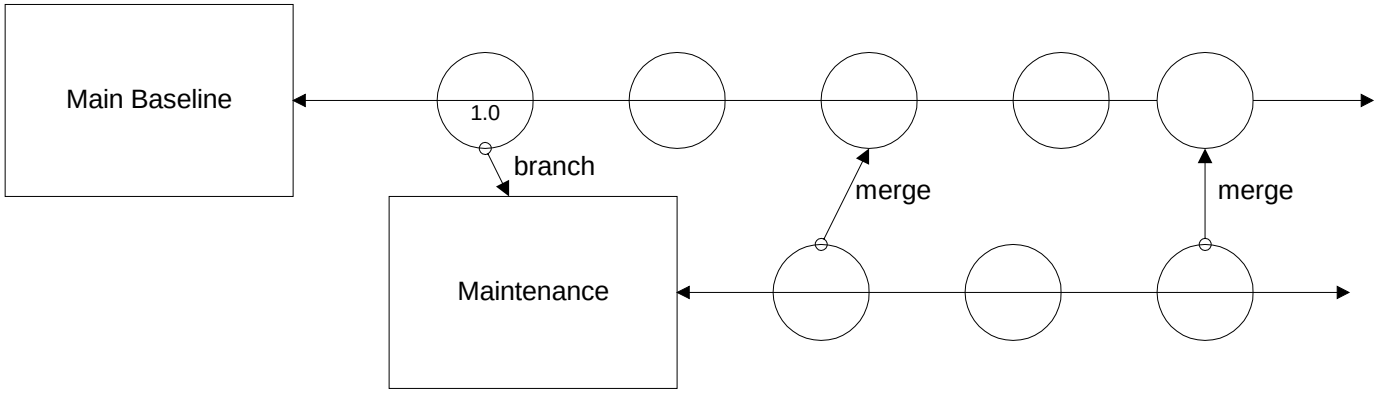
II. Bug Fixes

StarTeam offers users the ability to create Fix Branches (or Maintenance Branches) from the main development baseline for the purpose of creating service packs or other bug-fix releases. After the service pack is released, you can merge the Fix Branch back to the main baseline of development.

Sample Scenario 3

During the development cycle both Joyce and Mike have changes that need to be checked into StarTeam for Release 1. Joyce realizes that there are some bug fixes she needs to make from the main baseline. However, she does not want to affect development on that baseline. How should Joyce handle these bug fixes?

Figure 4: Parallel Maintenance and Development



Using a Fix or Maintenance Branch to Resolve Scenario 3

When maintenance and parallel development need to occur simultaneously, StarTeam recommends the use of branching and merging. The steps in this process are:

1. From the main baseline of development, derive a new view from the current view.
2. Permit items to branch and branch all items.
3. Follow the wizard. (That is, give the new view a name and a new working folder. Select a configuration based on the specific date and time, label, or promotion state).
4. Perform all of your bug fixes in the Maintenance View.
5. Upon completion of your bug fixes, select Compare/Merge… from the View menu to merge your items back into the main baseline of development.

III. Independent Product Development

StarTeam allows users to create a separate development baseline for independent development efforts. This independent development baseline can be based on the main baseline in its current or an earlier configuration.

Sample Scenario 4

Developers Joyce and Mike are developing code for Product A in the main baseline. Tim and Jill are developing new code for Product A1. Product A and Product A1 are independent products that have their own files, tasks, topics, change requests, and labels. Tim and Jill need to create a view (from the main baseline) that permits branching. For example, Product A might be Product A1 with a few special features for a specific(but not typical) audience.

Using a New Branch to Resolve Scenario 4

When developing an independent product based on an existing product, StarTeam recommends creating a branching view. Each item in the new view should branch when that item changes in the new view. The steps to ensure this are:

1. From the main baseline, derive a new view from the current view.
2. Permit items to branch and branch all items.
3. Follow the wizard. (That is, give the new view a name and a new working folder. Select a configuration based on the specific date and time, label, or promotion state).
4. Check out the files in the new view, change them as appropriate and check them back in.

IV. Common Component Library Development

Often companies have products with components that share libraries of common source code files. The common source code files need to be included in the build for each of the components that use them. Users will need to create a project to store only those common libraries. Breaking up a product into separate components greatly enhances the flexibility of each integrated component, which can be easily labeled, branched, and sent through it's own promotion model sequence independently of any other component in the product.

Figure 5: Project Hierarchies that Share Common Code Files

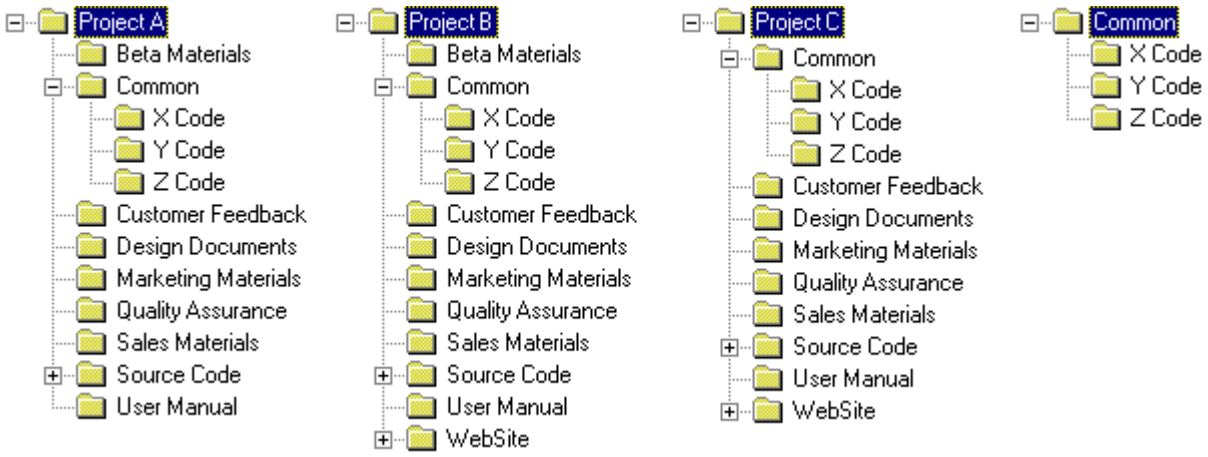


Figure 5 represents three independent StarTeam projects[1]. The Common project contains source code that is shared with each of the three projects (A - C)[2]. You can access the shared source code from any project.

You can set up the shared folders so that any change made to the source code in any of the shared projects or from the Common project immediately updates all the projects. You can also force changes made to the common code by project developers to remain only in the project in which the code was changed. You do this by not setting or by setting the Branch On Change behavior option, respectively.

[1] Note: These are projects and not views.

[2] Note: Users are not allowed to check-in files to shared folders. Additionally, if users want to delete a file from the common folder it does not ripple through to all the shared folders. The user needs to delete the file from each individual shared folder.

You can roll back the shared folders to use earlier versions of the common source code within a given project. However, you cannot roll the shared folders back based on a label or promotion state because separate projects (and views) do not share view labels or promotion states.

Projects A through C can be components of the same product or separate products, depending on your needs. The next few sections explain how to build Projects A through C (shown in Figure 5) at one time when they are components of one product. [3]

Setting up the Build Process

When each component is an independent StarTeam project, you can easily label, branch and promote items without affecting other components/products. However, some additional work is required to check out all the components to create the end product. The easiest way to combine these projects is to create a build process using the StarTeam command line.

Checking Out by Build Label

Figure 6 is an example of a batch file that generates a build of the product composed of separate components, each of which is stored in a different project. Each component/project is checked out based on its build label: Project A uses its Build 5 view label; Project B uses its Build 2 view label; and Project C uses its Build 3 view label. The check out process retrieves only the revisions of the files with the correct build label. All files from all projects are checked out to "C:\production build". This batch file compiles the projects and creates one single product. To reuse this build without having to remember what label was used for each component, check build.bat into StarTeam and assign it a build label. In the future, if you want to recreate this build, check out build.bat from StarTeam and execute it. Be aware that if you have shared folders of common source code that the following example checks that code out three times to three different locations. You may have to make allowances for this.

Figure 6: Build.bat

```
@ccho off
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project A/Project A" -is -x -q -o -fs -rp
"C:\Production build" -vl "Build 5" *
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project B/Project B" -is -x -q -o -fs -rp
"C:\Production build" -vl "Build 2" *
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project C/Project C" -is -x -q -o -fs -rp
"C:\Production build" -vl "Build 3" *
```

Checking Out by Promotion State

You can quickly create a current build of your product by checking out files by promotion state. The advantage is that you do not have to set up the items with the appropriate labels for each component. While this can be a great time saver, remember that promotion states float. So using this batch file to generate a build today could generate a different build tomorrow, depending on what labels have been assigned to which promotion states. Figure 7 demonstrates a three-step promotion model for each project. (For more information about setting up a promotion model, see Chapter 9: “Using Promotion States” in the StarTeam User's Guide.) Each promotion state is assigned a corresponding build label. To quickly check out the current production version of your product, run a batch file (see Figure 8: production.bat) that checks out all files from each sub-component currently in the Release promotion state.

[3] The user can build A and C and exclude B if needed. The development model allows the user to build interchangeable components to produce a product.

Figure 7: Three-step Promotion Model

	Project A	Project B	Project C
Release	Build 7	Build 4	Build 5
QA	Build 6	Build 3	Build 4
Testing	Build 5	Build 2	Build 3

Figure 8: Production.bat

```
@echo off
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project A/Project A" -is -x -q -o -fs -rp
"C:\Production build" -cfgp "Release" *
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project B/Project B" -is -x -q -o -fs -rp
"C:\Production build" -cfgp "Release" *
stcmd co -p "Administrator:Administrator@Whitestar:49201/Project C/Project C" -is -x -q -o -fs -rp
"C:\Production build" -cfgp "Release" *
```

Running production.bat produces an identical check out to build.bat. However, if the Release promotion state is assigned a new build label, then running production.bat will generate a new build and build.bat would not. To easily roll back to previous builds, it is important to construct and use both batch files. Check the .bat files into StarTeam for repeated use.

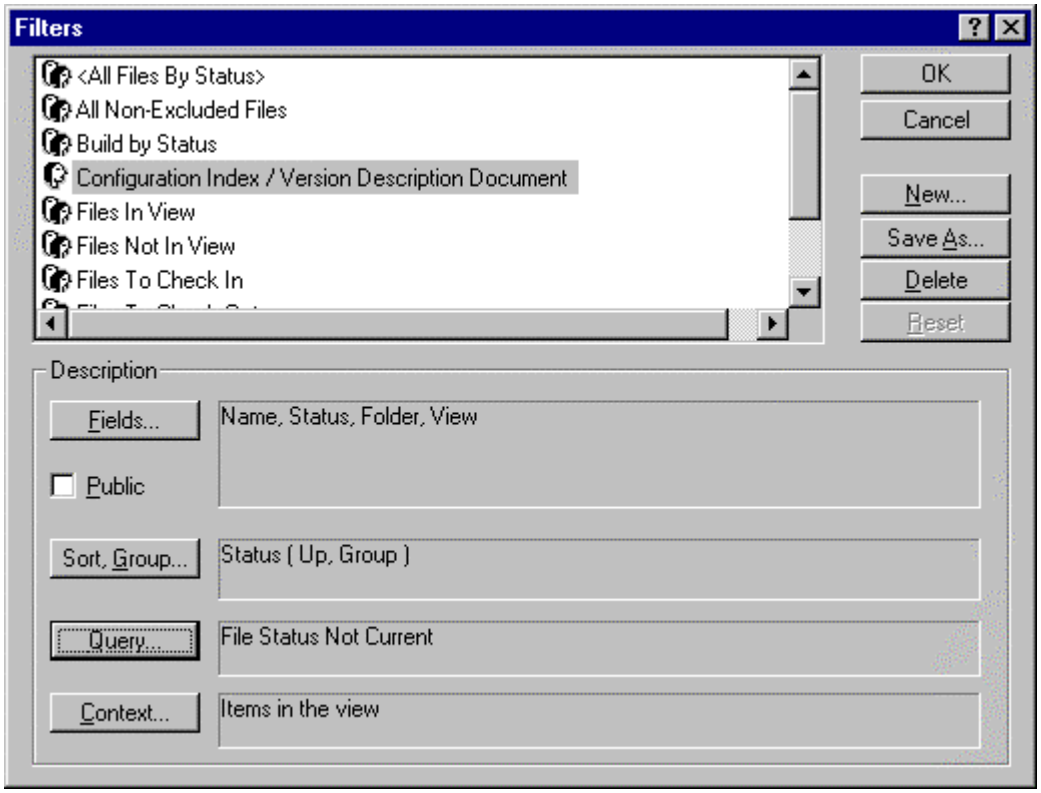
V. Configuration Identification Reports

StarTeam allows its customers to create configuration identification reports for builds and/or projects. A configuration identification report indicates what files have changed, been added, or been deleted since the previous build.

The following procedures walk you through the steps for creating such a report. You create a filter (which can be reused for all configuration identification reports), display the correct files in the files list, apply the filter, and generate the report.

Creating a Filter

The filter you create will be similar to the filter described in the following dialog.

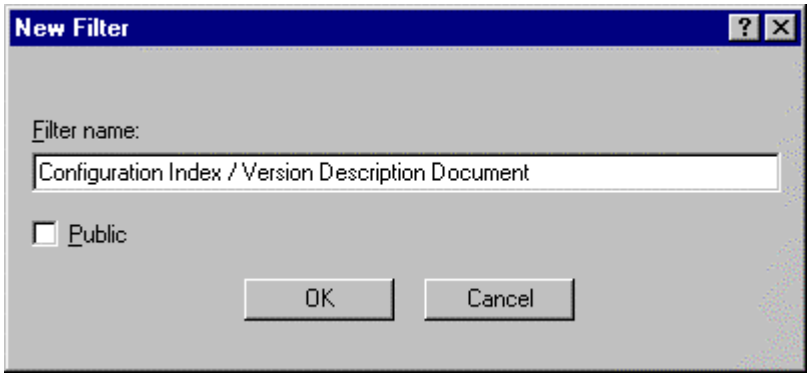


1. Create a filter named Configuration Index / Version Description Document.

   a. Select the Files tab to display the files list.

   b. Select **File⇨Filters⇨Filters**… to display the Filters dialog.

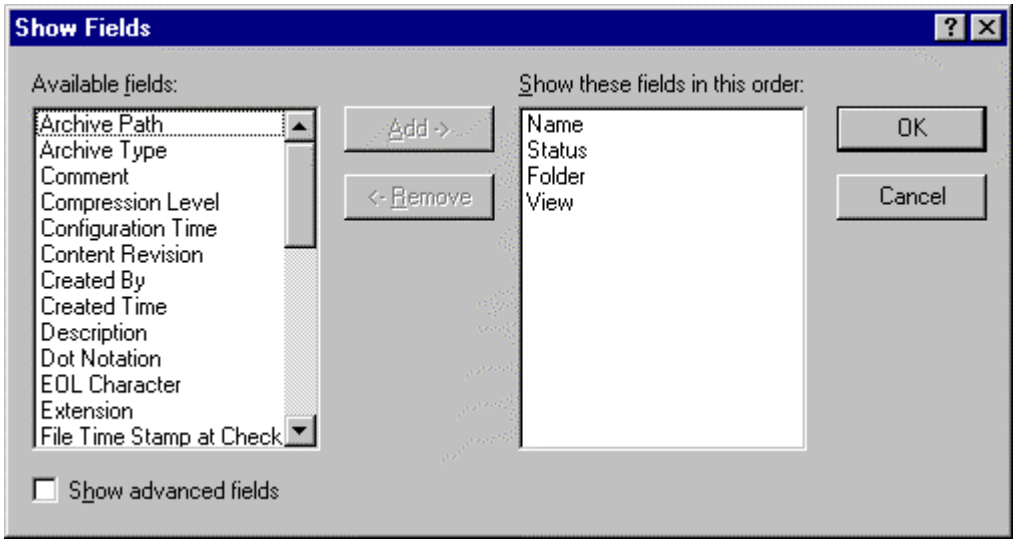
   c. Click New… to display the New Filters dialog.



   d. Name the filter; make it public if it will be used by more users than yourself; click OK.

2. Specify the fields to be displayed.

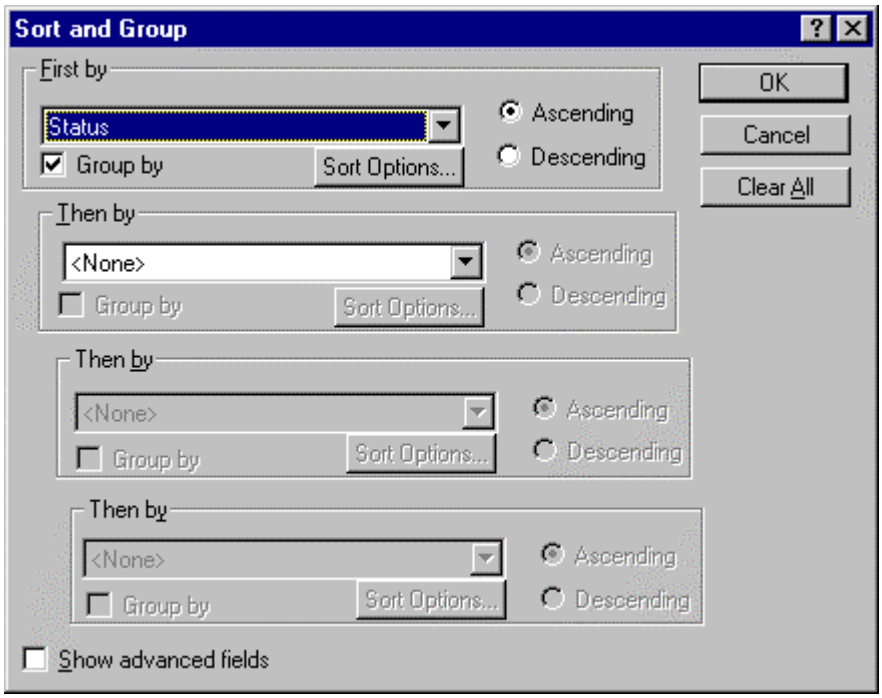
   a. Click Fields… to display the Show Fields dialog.



b. Select Name, Status, Folder, View, and any others of interest; click OK.

3. Sort and group by Status.

a. Click Sort, Group….to display the Sort and Group dialog.

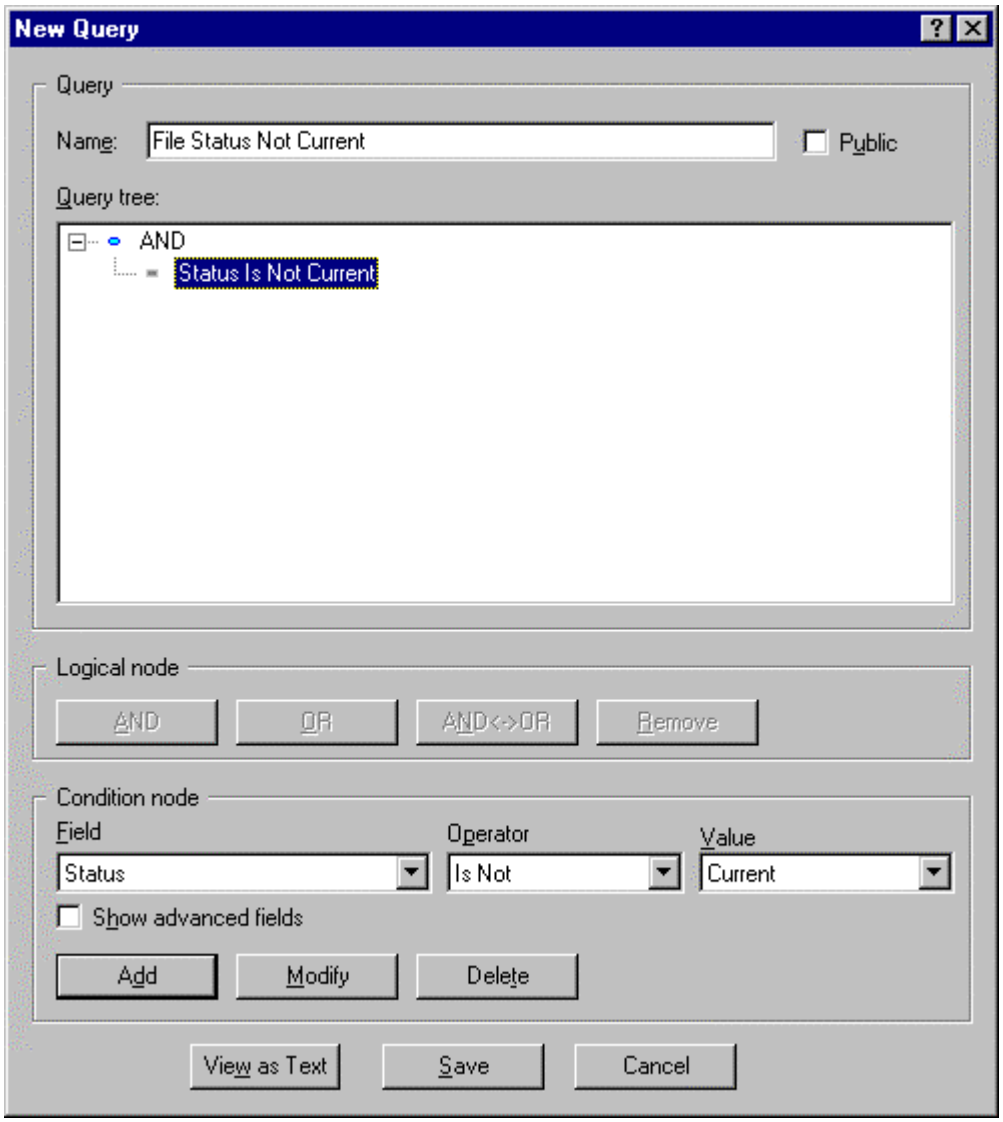


b. Sort and group the files by Status; click OK.

4. Create a query that will locate all files whose status is not current

a. Click Query to display the Queries dialog.

b. Click New… to display the New Query dialog.



c. Create a condition that locates all statuses except the current status; click OK.

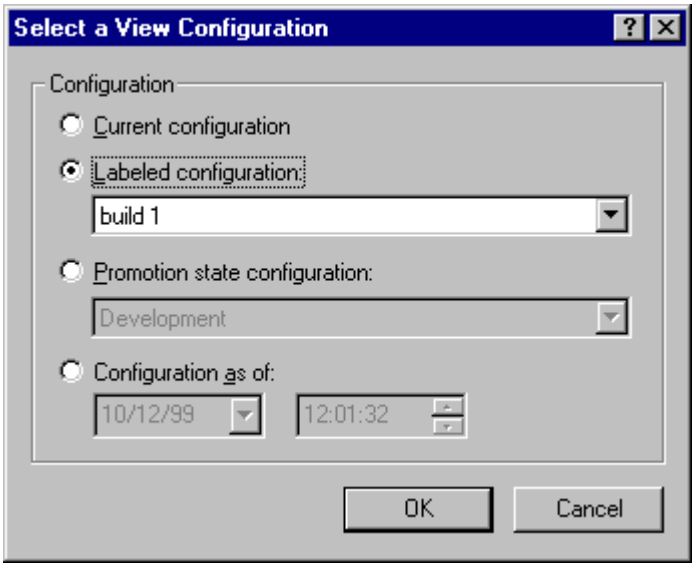
Displaying Modified Files and Generating the Report

1. Check out all the files based on a particular build (for example, build 2).
   NOTE: Delete all items from the directories/folders to be used in the check-out process prior to checking out the build.

2. Roll the view back based to the previous build label (for example, build 1).

   a. Select **View⇨Select Configuration…** to display the View Configuration dialog.

   b. Configure the view to the previous build label.

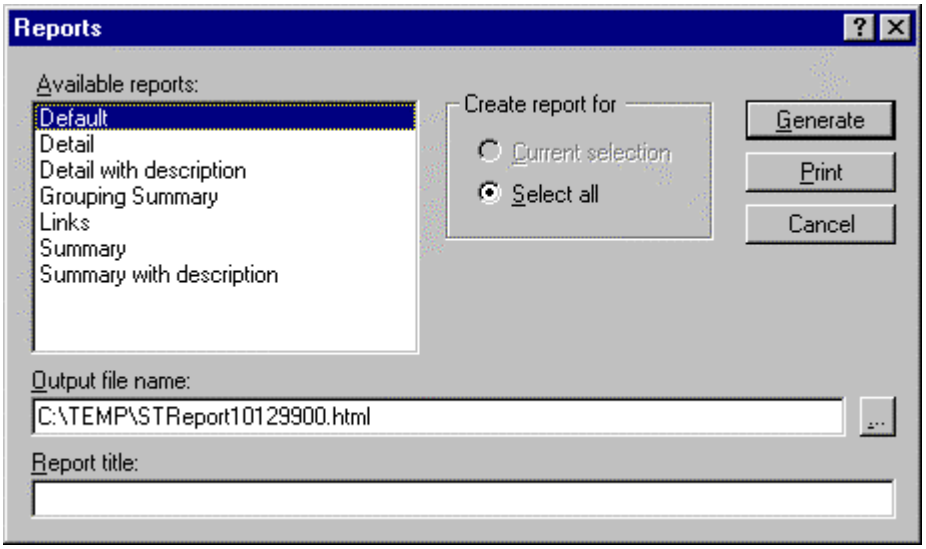


3. Some file statuses will change from Current to Modified for files that were modified in build 2, to Not In View for files added to build 2, and to Missing for files deleted from build 2.

4. Apply your new filter (Configuration Index / Version Description Document) to eliminate all files that still have the status current (if any) and to group the files for your report.

5. Select **File⇨Reports…** to display the Reports dialog.

6. Click Generate to create a default report.



VI. Staging

Many content managers use staging to assist in the software development life cycle (SDLC) process. For example, content managers usually have three stages: development, test, and production. They move content from one stage to another as it is developed, tested, and put onto the production server. This may involve managing three computers, one serving each stage of the process, with VSS installed on each computer.

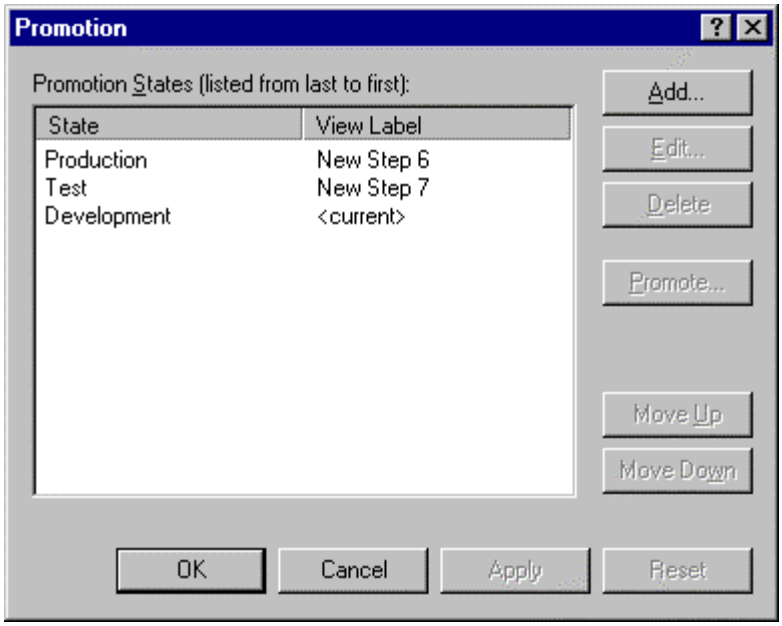
Instead of managing three computers with VSS installed on each, you can use StarTeam to manage one computer using one project with three StarTeam views (as explained below). This reduces the number of administrative and management tasks.

The three views are separated logically into different web root folders, and a single web server can reference them. The files can be checked out manually from each view to separate working folders/directories. This is acceptable practice for intranets or any inside-the-firewall web application. Because of the high traffic and external-to-the-firewall location of a consumer web site, a consumer web site's production server is often on a separate computer. However, StarTeam views still makes content management easier.

The following example has a phase for each stage. It uses the StarDraw sample project in which you create both views and promotion states.

Phase 1: Creating Promotion States for Your Stages

1. Open the StarDraw project to its root view.
2. Create the following promotion states in StarDraw (using **View⇨Promotion…**).
   - Production—Assign the label New Step 6 to this state.
   - Test— Assign the label New Step 7 to this state.
   - Development— Assign <current> to this state.



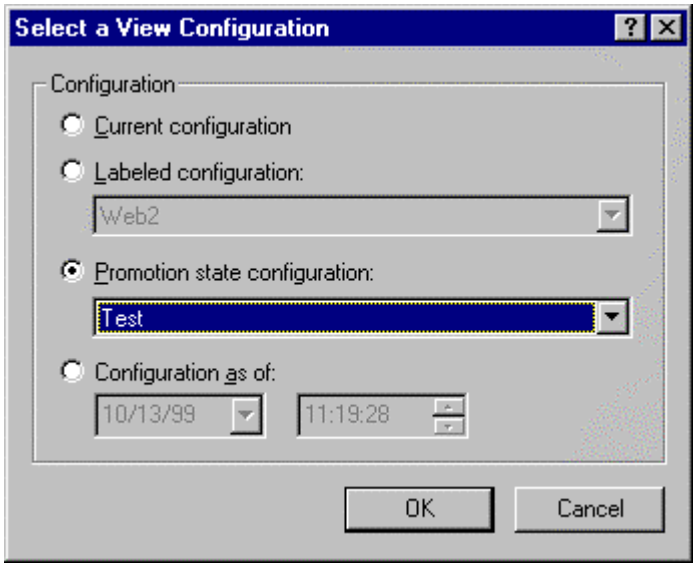
NOTE: As you create the promotion states, there will probably be states for which no view labels currently exist. Use <current> for these states. You can assign a view label to a state when the files associated with that view label meet the criteria required by the state. If you prefer, you can promote the view label from the preceding state to the label-less state.

TIPS:

- Use the Move Up and Move Down buttons to rearrange the states. They should be in order from last to first (final state down to initial state).
- Click Reset to reread the list of states from the server. This reverts to the states that were available when you opened this dialog or when you last clicked Apply.
- Click Edit... to change the selected state's name, description, or view label.
- Click Delete... to delete the selected state from the list.

Phase 2: Creating a View for the Test Stage

1. From StarDraw's root view, select **View⇨New…** and create a new view.
2. By default, the Derive New View From Current View check box is selected. Click Next.
3. Name this view Test then click Next.
4. By default, the StarDraw folder and all its child folders will be in the new view. Click Next.
5. Change the working folder/directory (just in case the testing group needs to check out some files). Click Finish.
6. Configure the view to show all the files, etc. with the promotion state Test:
   a. Select **View⇨Select Configuration…**.
   b. Select the Promotion State Configuration option button.
   c. Select the promotion state Test from the list box.



NOTE: You will need to set access rights to this view so that only the testing group can view and modify its change requests, tasks and topics.

Phase 3: Creating a View for the Production Stage

1. From StarDraw's root view, select **View⇨New…** and create a new view.
2. By default, the Derive New View From Current View check box is selected. Leave it selected.
3. Also select the Read-only check box. Click Next.
4. Name this view Production then click Next.
5. By default, the StarDraw folder and all its child folders will be in the new view. Click Next.
6. Change the working folder. Click Finish.
7. Configure the view to show all the files, etc. with the promotion state Production:
   a. Select **View⇨Select Configuration…**.
   b. Select the Promotion State Configuration option button.
   c. Select the promotion state Production from the list box.

NOTE: You will need to set certain rights to this view so that only the production group can view files.

Tips for StarTeam 4.1

StarTeam is very flexible and can fit many different work styles. You need to mold StarTeam to best meet your team's needs. Each possible project option has it pros and cons. This paper explains some basics that you need to consider. After reading it, you may want to call product support (at 714-445-4460) to go over some details. If you describe your product and work style, StarBase product specialists can recommend some StarTeam options over others.

Databases

Unless you have a special need for a particular database format, StarBase product specialists recommend using the database with which you are most familiar. If you are not familiar with any databases, they recommend using an MS Access database. StarBase stores its own files in StarTeam, using StarTeam VirtualTeam Server and an MS Access database with 50+ users. This MS Access database runs as reliably and as fast as most SQL databases. In addition, it is easier to administer. While there is a 1.2 gigabyte limitation on the size of MS Access databases, this limitation is rarely reached because StarTeam stores data files in a folder, known as the StarTeam vault or repository, rather than in any database.

Project Structure

StarBase product specialists recommend using one StarTeam project for each of your products. A project normally contains a product's source code, documentation, and other ancillary files. The least desirable layout places multiple products in a single StarTeam project, making it hard to version each product independently. When you create a view label in a project, that label tags every item in the current project view. (This includes StarTeam folders, files, change requests, and so on.) If you place multiple products in the same view, all of them will have the same labels. This can cause confusion when labeling builds and processing change requests.

Folders

In the best case scenario, each StarTeam folder and project view corresponds to a unique physical location, known as a working folder. The working folder stores checked-out files and is the location from which new files are added to StarTeam's revision control system. Avoid giving two or more StarTeam folders or views the same working folder (or folder structure)—even though StarTeam allows it. Having a one-to-many relationship between a working folder and StarTeam folders leads to file status problems.

If you need to check out many files from many folders in StarTeam to one common physical folder (for example: to use in a build environment), you should use the command line to do so. Use the /fs parameter to ignore file status as well. This will save a lot of time.

Backups

Completely back up your project on a regular basis. Be sure to back up all the necessary files and folders at the same time. The StarTeam Administrator's Guide (Appendix C, "Backing Up StarTeam") describes the proper backup process. However, it cannot be emphasized enough that the database and the StarTeam repository must be backed up in synchronization. If the StarTeam file repository and the StarTeam database are backed up at different times, and users have accessed StarTeam between those two times, the repository and the database will not be synchronized. This is a situation which can easily occur using databases such as MS SQL or Oracle, which often times run on completely separate machines than the StarTeam VirtualTeam Server, and may be on their own backup schedules. Restoring out-of-sync repositories and databases will cause operational problems in StarTeam. Even though there is a vault verification utility that can be run to resynchronize the database with the repository, there will be some inevitable data loss. To avoid this scenario, you should do one of the following to ensure a synchronized backup:

- Lock the server for the duration of the entire backup using either the StarTeam command line or server administration tools
- Shut down StarTeam VirtualTeam Server

Security

By default, all users have access to everything in StarTeam. This would include deleting the project. To avoid accidental deletion, set up access rights as soon as possible.

Things to remember about security include:

- StarTeam checks for access rights from the lowest level (the item level) to the highest level (the project level). For example, if you set access rights on an individual file, then folder, view, and project level access rights are never checked for that file.

- If you set access rights for any user or group on a tab in an Access Rights dialog, all users **not** included on that tab are denied **all** access rights at that level for that tab.

  In general, if you set access rights of any kind on any tab, you must be sure to set the access rights for all users and/or groups that need to access the objects affected by that tab.

- Access rights can be overridden by:

  - The fact that a user is the object's owner. Usually, the owner is the person who created the object. Ownership information is stored in the server configuration's database but is not displayed in StarTeam.

    In StarTeam 4.2, this override can be ignored.

  - Privileges given to a group that includes the user. These are set per group from StarTeam VirtualTeam Server.

    In StarTeam 4.2, this override can be ignored.

    By default, the Administrator group has full privileges (rights to do anything and everything). The Security Administrator and All Users groups have no privileges.

- When a user belongs to several groups, that user has a right if it is granted by any of the groups. In other words, a user has the maximum amount of access allowed by the combined groups in which he or she is a member.

- Access rights are **not** inherited. However, they remain with a folder or item until it branches.

  Every view in a project has the same project-level access rights. As you derive a branching view from an existing view, the child view does not inherit access rights from its parent. However, each folder or item in the child view that existed in the old view has the same folder-level or item-level access rights that it had in the parent view—but only until that folder or item branches. Then that folder or item no longer has any access rights at the folder or item level. If you have the right to view access rights, you can see these pre-branching folder-level or item-level rights from both the parent and child views. Changing these rights in either view changes them in the other view as well because you are changing the rights on the same folder or item.

- Access rights go with moved or shared folders or items.

  As folders are moved or shared from one view to another, they take any access rights assigned to them at the folder level to the new view. Similarly, items (files, change requests, tasks, and topics) that are moved or shared from one view to another take any access rights assigned to them at the item level to the new view. Shared folders and/or items can be branched, at which time they lose their access rights.

From StarTeam VirtualTeam Server

On StarTeam VirtualTeam Server, you use User Manager to create users and groups for each server configuration (while that configuration is running). Use the following guidelines:

- Do not change the privileges for the All Users group or the Administrators group. Be aware that these group privileges override any security set to the contrary at the project level.
- Do not create additional groups under the Administrators group, as they will inherit the Administrator's groups privileges and become difficult to secure.
- Create the groups that you need under All Users or under each other. For example, you may need to create the following or similar groups: Developers, QA, and Documentation.
- Create users and assign them to groups. Make sure that at least two users are administrators, in case one administrator is locked out for some reason.

From StarTeam

From StarTeam, you can set access rights:

- At the project level by selecting **Project⇨Access Rights…** from the menu bar.
- At the view level by selecting **View⇨Access Rights…** from the menu bar.
- At the folder level by selecting **Folder⇨Advanced⇨Access Rights…** from the menu bar.
- At individual item levels by selecting **<item>⇨Advanced⇨Access Rights…** from the menu bar (Items are files, change requests, tasks, and topics.) It is unusual to set rights at this level, but you can do so if you choose.

Use the following guidelines:

- Set up security at the project level first.
  - Set it for every group (except the All Users group) on every tab. (The tabs are Project, Views, Child Folders, Files, Change Requests, Tasks, and Topics).
- Set up security at the view level.
  - If you set access rights for a tab, remember that any user that does not have rights on this tab is denied all rights at this level for this tab. If you set access rights on a tab, you must set rights for every user and/or group that needs access at this level for this tab. (The tabs are Views, Child Folders, Files, Change Requests, Tasks, and Topics). This applies to the folder levels and item levels as well.
- Set up security at the folder level only if you really need to.

  Remember that these settings go with the folder as it is moved or shared and as it becomes part of new views (until the folder branches in the new view). Also remember that folders branch only when their properties change, and that their properties tend to change infrequently.
- Avoid setting up security on an item.

Remember that, if views derived from this view at a later time include this item, these settings stay with that item until it branches. Also remember that these settings go with the item if it is moved or shared.

- You can deny rights as well as grant them, but it is best to grant them only.

  If you deny rights, be sure to observe both of the following:

- **Always** make sure that the deny rights records on any tab precede any records that grant rights on that tab.

- **Never** allow any tab on an Access Rights dialog to have only deny rights records.

Manipulating Folders and Items in StarTeam

Folders and items (files, change requests, tasks, and topics) can be moved, shared, and deleted. They can be moved and shared within the same view or between views—so long as the views belong to projects that use the same server configuration.

Some of the complications that arise from moving a folder or item are:

- If you move a folder or item outside of a view and then roll the view back (to a date when the folder or item existed in that view), it no longer appears. This is because the moved folder or item is no longer attached to that view. This is not true of deleted or shared folders and items.

  Deleting folders and items does not decrease the size of your repository or increase performance. Deletion does not really delete anything; it is more of a retire function. The folder or item is visible only when you roll back the view to a time before the deletion.

  When a folder or item becomes obsolete, consider moving it to another folder in the same view (perhaps named Obsolete) or deleting it.

  If you need an item in another view, consider sharing it with the other view.

- If you move a folder or item from one view to another, any labels attached to it do **not** go to the new view with it. Moving within a single view leaves the labels intact.

- If you move a change request from one view to another, the workflow processing for it can be affected as follows:

  - If the Last Build Tested and the Addressed In Build fields in a change request have build labels as their values (in other words, these fields are not empty and do not contain the value “Next Build”), the moved change request retains those values.

    In the new view, these values can be changed, but only to the names of build labels that exist in the new view.

  - If the Addressed In Build field contains the value “Next Build” at the time of the move, the “Next Build” value is replaced by the name of the next build label to be created in the original view—not the next build label created in the new view. This is true even if other changes have been made to this change request while in the new view.

  - If a change request’s Last Build Tested and the Addressed In Build fields have no values at the time of the move, the change request’s workflow is specific to the new view only.

  The comments about workflow in this section also apply to change requests that appear in both a parent view and the branching view derived from it. However, the workflow is affected by a change request’s values in the

Last Build Tested and the Addressed In Build fields at the time that the change request branches—rather than at the time it is moved.

When you merge change requests, you can control which properties end up in which views. However, when you merge change requests, you can expect the same complications explained above. For example, the next view label in the view where the "Next Build" value was assigned becomes the value of the Addressed In Build field, regardless of the view in which the merged change request now resides.

Sharing a folder or item can result in complications as well:

- Circular shares are to be avoided. A circular share is one in which a folder or item, such as a file, is shared back into the original location. For example, you might share the same folder from the first view to a second view, from the second view to a third view and from the third view back into the first view. The share back to the first view makes this example a circular share.

- Deleting the original of a shared folder or item does **not** orphan the shared ones. Remember that nothing is ever really deleted from StarTeam.

- Avoid redundant shares. A redundant share is one that is identical to another share. For example, if you share the same folder from the first view into the second view more than once, you create redundant shares. Worse, the identical shared folders have files with the same names stored in the same working folders. Checking files out from one folder overwrites the files from the other and results in file status errors.

- If you share a folder or item into a reference view, you are really sharing it into the parent view. If the parent view was the source of the reference view, this is another example of a redundant share.

- If more than one third of a project consists of folders or items shared into the project, you should probably rethink the project's structure. It may be more complicated than necessary. Also, large amounts of sharing can cause some performance problems.

Labels

StarTeam has two types of labels: view labels and revision labels. A view label belongs to the view in which it is created and has no meaning outside of that view. It is a point-in-time marker that StarTeam applies to every folder and item in the view at the time the label is created. Usually, you create a view label when you want to take a snapshot of a particular view and a particular point in time. Rolling back a view based on that view label as an easy way to get back to that important time in the past. This is not an entirely accurate description, because view labels are a little more versatile then this, but this conceptually describes the purpose of a view label.

A revision label allows you to identify a specific revision of an item or specific revisions of a set of items. The most common use of a revision label is to group a subset of files within a view for the purpose of checking them out. A revision labels can be moved from one view or project to another, but only if you clone the label. In general, revision labels are part of a view and do not move with items to new views.

If you create a reference view, the view and revision labels are still available in the new view, because a reference view is just a different way of looking at the same view rather than a different view. However, if you create a branching view, none of the labels from the parent view are automatically in the new view.

Also, if you move or share an item, such as a file, from one view to another, the labels attached to that item do **not** go with it. Labels created in the items' new view can be attached to moved items, but not to shared items unless the shared items have been branched within the new view.

View Labels

Within a project, set up the branching views so that they contain all the items that you wanted tagged with the same view labels. If you don't want everything to receive a given view label, then those components should be in another view.

The most common use of view labels is for labeling important dates or even milestones in your project. For example, you can use a view label to indicate what files were included in a build of your project. The label's name might be "Build 425." After creating a build label, the files with that label are checked out for the build. This prevents the build from including unknown files that were added or checked in at the last minute.

You can add revisions to or remove revisions from the view label after the fact. This allows you to add files to a label that didn't exist in the project when the label was created and remove files from a label that don't logically belong in the labeled set—even though they were present in the view when the label was created. When a view is rolled back by the label, it then shows only the files to which the label is attached, giving a slightly adjusted picture of the view. Such adjustments can be very useful, even if the view label no longer represents an exact point in time.

StarTeam lets you roll a view back to any point in the past, based on a specific time, label, or promotion state. For example, you can roll back the view to the label named "Build 425." Rolling back to a specific label can be more useful than rolling back to a specific time, because the label might have been adjusted.

Build Labels and Change Requests

View labels can be designated as build labels. Build labels differ from other view labels in only one respect: they affect the workflow of change requests. When you create a build label, StarTeam searches for all change requests with the value "Next Build" in the Addressed In Build field and changes the value "Next Build" to the name of the newly created build label. This lets testers know what build to test for the change requested in the change request.

See the section titled "Change Requests"for details about what happens to change request workflow as new branching views are created or as change requests are moved or merged.

View Labels and Project Progress

You can use view labels to show a project's progress. For example, you can create a view label when the code is ready for the next stage of development. For example, you might name a view label Test or Production. Better yet, you can use promotion states with your view labels. See the section titled "Promotion States."

Usage Problems

Problem: Some users try to extend the functionality of view labels beyond the use for which they were designed. For example, suppose you want to label only one branch of the StarTeam folder hierarchy. Some users create a **view label** that affects the entire view. Then they either delete it from the root folder and add it back to a specific branch or they delete that label from several of the branches. This is **not** a good idea. When you create a view label and then delete it, the program creates a huge exclusion list saying everything is labeled except these items, which happens to be most or all of the view. The big exclusion list can cause performance problems.

Solution: If you need to be label specific branches of a view independently from the rest, that branch might be better off in its own branching view. As an alternative, you can use a revision label to label only one branch or a few branches of the StarTeam folder hierarchy. You cannot roll a view back to a revision label, but you can select all the files, for example, with that label or check all of them out based on that label.

Revision Labels

The most common use of revision labels is to help developers locate a subset of file revisions that either:

- Are members of the same logical grouping but are in a variety of folders
- Are not the only files in a given folder, but need to be isolated quickly for special builds

The revision label makes it possible to check out all of the labeled files at one time, regardless of where the files may have been moved within the view.

Revision labels can be used with items other than files. For example, a tester might label a small group of change requests and later select them by label.

Usage Problems

Users tend to think that a revision label is a unique way of identifying a folder or an item, such as a file, forever. They think that the label stays attached to the folder or item, even if they move the item to another view or project. StarTeam does not work this way. The revision label remains attached to a specific revision of a folder or an item as long as the folder or item remains in the same view. However, a user (with the correct access rights) can detach the label from the item or attach it to a different revision.

Be aware that a revision is attached to a specific revision of a folder or item. For example, if a file has ten revisions, the revision label would be attached to only one of them, such as Revision 5. When you check this file out based on its revision label, you check out Revision 5. Other configuration management applications check out the most recent revision of a labeled file. This can confuse users who have experience with other applications.

Promotion States

Most products go through some sort of release or production cycle. Using a very simple example, a software product cycles from developers to testers to the marketplace. Normally software files are continually added to and corrected as the testers notify the developers about flaws in the product. Finally, some build or version of the software is released for general use.

A promotion state groups items, usually files, that have reached a certain stage of development. It provides a convenient mechanism for ensuring that the right files (or other items) are available to the right people at the right time. For example, if a software administrator creates the promotion states Test and Release, files that are being worked on by testers can be assigned to the Test state and those that are ready for release can be assigned to the Release state.

Each promotion state is based on a view label, which is usually (but not necessarily) a build label. A view can be rolled back to the promotion state, so that users work with the file revisions that are at the correct stage of development. The label associated with a promotion state is changed when appropriate. For example, one week Build 07 may be assigned to the Test state, and the next week, Build 08 may be. The advantage to using a promotion state is that the user always uses the same promotion state and doesn't need to be concerned with a specific view label.

The promotional model as a conceptual practice and the promotion states implemented in StarTeam are different. The StarTeam promotion state feature permits an administrator to create promotion states and associate a view label with each. The view label for a state can be changed whenever appropriate. It can also be promoted from one state to the succeeding state. For example, while testers may always

be using files in the Test promotion state, these may be the files from Build 07 one week and the files from Build 08 the next.
Users usually configure a project view for their job assignment by promotion state instead of by view label. For example, testers would configure the view to the Test promotion state.

Change Requests

A change request either requests a fix for a defect or suggests an enhancement to a product. These may come from users who call product support or from in-house testers. A change request is not a work order. It is only a note explaining a perceived problem. Usually you do both of the following:

- Provide all employees with guidelines about how to write a change request.
- Set up a system by which the change requests are prioritized and marked as "Open," "Deferred," or "As Designed." "Open" change requests are assigned to team members for implementation.

You **cannot** customize the workflow for a change request. When you submit a change request it defaults to the "New" status. You can change its status to "Open," "Fixed," "As designed," "In Progress," "Documented," etc. From a given status, you can only change to a few possible next statuses. For example, while a change request's status is "Fixed," you can only change its status to "Open" or "Verified Fixed." From there you can only change it to "Open" or "Closed." The status can't go from "Fixed" to "New."

If you have StarTeam Enterprise, you can create a custom field to replace the current Status field. While the new field cannot control change request workflow, you can use if for sorting and reporting. You can also edit the Status field to change the name of the possible statuses. For example, you might want to change "New" to "Entrant" or change "Closed" to "Finalized." However, you cannot reorder or disable any status values.

Topics

StarTeam associates threaded conversations with folders in the StarTeam folder hierarchy. Topics can raise general questions about the project or start very specific discussions about issues, such as feature implementation. While the responses can lead to resolution of these issues, the historical value of these conversations to the project can be even more significant. Future team members can:

- Reassess decisions more capably
- Avoid retrying solutions that were previously found faulty
- Understand why a particular solution to a problem became necessary and, therefore, not replace that solution with one that doesn't meet all the necessary criteria

Similar to a newsgroup, topics provide a forum for discussion. Everyone involved in the project can collaborate without using untraceable and unthreaded e-mail messages. Topics reside in one area that is versioned and centralized. After a discussion of the pros and cons etc., tasks can be assigned.

Tasks

StarTeam's task component is available only with StarTeam Enterprise. It allows local and remote users to report their efforts on the tasks that they have been assigned. The Task component can interoperate with MS Project and the rest of the StarTeam components or solely with the StarTeam components. However, you use the Task component very differently in each case.

When the Task component interoperates with MS Project, some of the usual task operations can be performed only from MS Project.

Glossary

Access Rights

A security feature. The rights granted (or denied in exceptional cases) to users or groups that allow team members to see items, modify them, create them, and so on.

All Descendants

The button at the top of the right pane. Also a command on the File, Change Request, Task, Topic and Audit menus. When selected, the view window displays information for the selected folder and its child folders. Otherwise, the view window displays only the items associated with the folder and not with its child folders.

Alternate View

A view derived from the main or default view.

Alternate Working Folder

Creating an alternate working folder allows you to store that folder's working files on your workstation at the location you specify. Creating an alternate working folder for the root of a StarTeam view or a branch in a StarTeam folder hierarchy can alter the paths of the working folders for child folders.

Archive

In version control, the file or group of files that make it possible to recreate past revisions of a file that is under version control. An archive can also be, as in StarTeam, the folder that stores such files.

Audit Log

A chronological record kept by StarTeam showing all actions performed on StarTeam folders, files, change requests, tasks, topics and responses.

Branching

The process of creating an independent item that is derived from a corresponding item in a parent view. In the case of a text file, the branched item can later be merged with the file 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 StarTeam folder hierarchy or a topic tree.

Branch on Change

Advanced field. Enumerated type. Indicates whether a file will branch when it changes. The values are No and Yes.

The value is No if the file's behavior is not set to "Branch On Change." (Perhaps the file is in the root or a reference view and the "Branch On Change" feature is disabled. Perhaps the file is in a variant view but has al ready branched as a result of a change, which, in turn, results in the "Branch On Change" feature becoming disabled. Perhaps the file is in a variant view, but its behavior currently does not permit it to branch on change. (That means that modifications are checked into the parent view.) If the value is No, the value of the Branch State explains the No.

The value Yes indicates that the file resides in a variant view, has its behavior set to "Branch On Change," but has yet to be changed.

Branch Revision

A special form of revision number that indicates the branch path for this revision.

Change Request

The list of change requests, related to your selection from the StarTeam folder hierarchy, that is displayed when you select the Change Request tab. The list is further refined by the All Descendants button and filter you select.

Check-in

The operation performed on a revised file that stores the new file revision in the archive (or vault) and data about that file in the repository.

StarTeam 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 releases the file to others by unlocking it.

Check-out

The operation that copies a revision of a file from the StarTeam project archive (or vault) to a team member's working folder. A team member can check out a file with or without the intention to alter that file. StarTeam 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.

Command line

StarTeam's command-line interface is the same for

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

Component

This document refers to parts of StarTeam as components. For example, it references the Task component. The files, change requests, etc. managed by the component are referred to as items.

Compression

Data that is transferred between your workstation and the server can be compressed. Data compression reduces the amount of traffic on the network. However, the time to compress and decompress the data is added to the transfer time.

Configuration

A relative arrangement of parts or elements.

StarTeam has view, folder, item and server configurations. A view, folder, or item configuration is the isolation of that view, folder, or item to a particular revision based on a point in time. For example, you can configure a view to:

Be current (so that you always see the latest revisions of every folder and item in the view).

A view label (so that you see all the revisions in the view that have the selected label assigned to them). A view label initially represents a point in time although the label can be adjusted to include revisions that were not current at that point in time and exclude revisions that were.

A promotion state (so that you can see all the revisions in the view that have been assigned the label that is currently associated with the selected promotion state).

Any selected date and time (so you call see all the revisions in the view that were current at the specified date and time).

You can also configure individual folders and items.

Current

File status. The content of the file in the working folder is the same as the content of the latest revision of this file.

Default View

Initial or main view for the project that contains the configuration used for primary development.

Defect

A fault or error in a product. Add and track defects using StarTeam's Change Request component.

Encryption

Data that is transferred between your workstation and the server can be encrypted. Encryption protects files and other project information from

being read by unauthorized parties over unsecured network lines (such as the Internet).

File Status

When you update the status of a file, StarTeam compares the working file with the revision you checked out and the most recent revision. For example, the file list may say that the file is Current, but someone else has just checked in a copy of it, so your status really is Out Of Date.

Updating file statuses is not the same as updating files. For example, if a file is not in your working folder, updating the status will let you know that the file's status is Missing. It will not check out the file for you so that it is no longer missing. After all, you may not want the file on your hard drive. Normally, you use a file's status to determine whether the file should be checked in, checked out, added, or ignored. For example, you may want to:

- Check in a file if its status is Out Of Date, Missing, or Merge
- Check out a file if its status is Modified or Merge
- Add a file to StarTeam if its status is Not In View

File-oriented version control

SCM applications where file revisions are stored in a specific archive file and is independent of the location where the file is placed when building the application.

Folder

StarTeam folders help organize the project view into discrete understandable parts. For example, a project for a software product might have SourceCode, User Manuals, and Corporate Libraries as folders. Each folder has a working folder that corresponds to it and exists on your hard drive. The StarTeam folder might have child folders. It probably has files, change requests, tasks, and topics associated with it.

Freeze/Frozen

An item is said to be frozen (and, therefore, read-only) if it is based on the state of its corresponding item in the parent view at a specific moment in time AND cannot be branched.

An item is also frozen if you reconfigure it to a specific label, promotion state, or time in its past.

Hierarchy

The hierarchical display of a StarTeam view and its associated folders. The folder hierarchy is always displayed in the left pane of the view window.

IDE

Acronym for integrated development environments such as PowerBuilder, Microsoft's Developer Studio, Oracle, etc.

IPX/SPX

IPX (Internetwork Packet Exchange) is a network protocol that interconnects networks that use Novell's NetWare clients and servers. IPX is a datagram or packet protocol. IPX works at the network layer of communication protocols and is connectionless (that is, it doesn't require that a connection be set up before packets are sent to a destination. Packet acknowledgment is managed by the Sequenced Packet Exchange (SPX).

Item

A StarTeam object or element. Items include projects, views, folders, files, change requests, tasks, topics, responses, and audit entries.

Labels

The act of attaching a view or revision label for one or more folders and/or items.

Labeled Configuration

The basis for a new view or a reconfigured view. The view contains only items with the label you specify. This option is disabled in the new view's parent view or the reconfigured view has no labels defined for it.

Merge

The file status that indicates that the working file is not based on the latest revision of the file. As you check this file in or out, StarTeam asks if you want to merge it with the latest revision.

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 StarTeam, when you reach a milestone, you can apply a view label, usually a build label, to indicate that the milestone has been reached.

Missing

File status. The file is not in your working folder. You might want to check the file out.

Modified

File status. The working file has been altered and is based on the latest StarTeam revision of this file.

	You might want to check this file in.
Named Pipe	An interprocess control (IPC) protocol for exchanging information between two applications, possibly running on different computers in a network. Named Pipes are supported by a number of network operating systems.
NetBIOS/NetBEUI	A common network protocol for PC LANs that provide session and transfer services. NetBEUI is an acronym for NetBIOS Extended User Interface and provides a standardized transport frame for NetBIOS.
Not In View	File status. The file is in the working folder, but is not in the StarTeam view. You might want to add this file to the view.
Object-oriented	StarTeam is an object-oriented SCM application. The StarTeam repository is an object-oriented data store that supports object versioning, linking and configurations.
Out Of Date	File status. The working file is a copy of an old revision of the file. If you need the current revision, you should check it out.
Pinning	You can link specific revisions of the linked items. The revisions selected for both items appear as columns in the link pane.
Project	A set of related files, change requests, tasks, and topics comprising a single product under StarTeam version control.
Project-oriented version control	SCM application that includes version control of all items (change requests, files, topics, tasks, folders) in a project. A project-oriented system allows users to view these items in different ways depending on their role or the immediate work requirements of the project. The project-oriented approach is a *superset* of the features found in the products that implement the file-oriented approach to configuration management.
Promotion Model	StarTeam provides an object-oriented architecture that supports entity-relationship modeling. StarTeam allows you to move (promote) changes between different stages of the project, for example from development to testing to product release, etc. Developers work on code changes in promotional states that are isolated from other development

efforts.

Promotion State

A state through which a product passes. For example, a software application goes though a development, test, and release cycle could use the promotion states Development, Test, and Release. In StarTeam, each promotion state has a view label assigned to it. The view label can change over time, but testers, for example, always working in the Test state, can be oblivious to what label is currently assigned to that state.

Promotion State Configuration

The basis for a new view or a reconfigured view. The view contains only items with the promotion state you specify. This option is disabled in the new view's parent view or the reconfigured view has no promotion states defined for it.

Read-only View

A reference view that cannot be modified from the reference view—although it may be modified as the parent view is updated. If it floats, it is updated. If it is based on a label (or a promotion state) and the items with the specified label change, the read-only reference view with reflect that. It is based on a specific date and time; it is frozen as a copy of what the parent view looked like at that point in time.

Reconfiguring

You can reconfigure a view, file, change request, topic, or task to a point in the past, defined by a label, promotion state, or a point in time.

Reference Count

A list of the items that reference another item. For example, a file may be shared by two project views on the same server (or even between folders in the same view) and, therefore, have two references to it.

Reference View

A view derived from a parent view that, in general, use a different StarTeam folder as the root folder and uses the same working folders at the parent. If it floats, it receives updates as the parent view changes. If it floats and is not read-only, it sends updates to the parent view as it changes. If the reference view is based a specific label or date and time within the parent view, it is frozen at that moment in time and is read-only.

Repository

The database and the archived files associated with a particular server configuration.

Revision

As an item, such as a change request is revised, each set of changes is saved as a revision.

Revision Labels

A revision label provides a convenient method of

identifying a revision of an item or a set of revisions by name. This is primarily used for files. For example, when you check in a group of files that may need to be checked out together, you can give them a revision label.

Roll Back

You can roll back a view to a past state based on a label, promotion state, or a point in time. For example, you might want to:

- Take a quick look at how things were when the Beta3 label was applied
- Recover an item that has been deleted by rolling back the view to a date before the item was deleted

However, this “freezes” the view until you change its configuration back to current or close the project (which automatically changes the configuration back to current). You cannot check in files, update change requests, etc. because you cannot change the past.

Root Folder

The topmost folder in the StarTeam folder hierarchy. It has the same name as the view.

SCM

Acronym for software configuration management.

Server Configuration

Contains the repository and option settings you select when you set up your StarTeam VirtualTeam Server. For example, the administrator may want projects to use encryption and compression, so the server configuration specifies these features. StarTeam items, such as folders and files, can be shared provided their projects us the same server configuration. A server can be started with any one of several server configurations, but that configuration controls what projects, etc. can be access during that session.

StarDisk

StarDisk is a virtual file system that allows you to use conventional Windows applications, such as Windows Explorer, Microsoft Word for Windows, and Microsoft Developer Studio, to access and manage files that are under version control. You use StarDisk to map a StarTeam view to a virtual drive. Then you can access any file on that drive from Explorer or another application. If the file is not checked out, StarDisk can check it out for you.

StarTeam File

A file under StarTeam version control; therefore, a file that is in a StarTeam project.

Storage Method

The revisions stored in the archive for a specific file can be changed from one type of storage and

compression to another. There are two types of storage: forward delta which is recommended for text files and full revision storage which is recommended for binary files and text files (like .rtf files) that change massively from revision to revision.

TCP/IP

A protocol for communication between computers used by the Internet. Acronym for Transmission Control Protocol/Internet Protocol.

Task Component

The task component is available with StarTeam Enterprise. StarTeam allows users to create, track, and resolve tasks related to the project. This component also interoperates with Microsoft Project 98.

Time Stamp

Information maintained by StarTeam about files and revisions.

For file revisions: The date and time that the file was checked into StarTeam.

For files: The date and time for the working file.

Tip Revision

The most recent revision to an item such as a change request.

Topics

The first message on a particular subject attached to a folder in the StarTeam folder hierarchy. Once submitted by one team member, others may respond to it, creating a topic tree.

Unknown

File status. The file in the working folder has the same name as a file in the view but the file was not checked out from the repository. You might have copied it from another location. Use Update Status to determine the correct status.

User

An individual given access to a server configuration and the projects it manages via StarTeam VirtualTeam Server. Usually, that access is protected by password. A user is also referred to as a team member.

Variant View

A view that may or may not be derived from a parent view. When not derived from a parent view it is a blank variant view. Variant views always permit branching. If they float and have the Branch On Change option set, they are updated by the parent view on a file-by-file basis until that file changes in the variant view. If they float and do not have the Branch On Change option set, updates are sent to the parent view until a point in time when the Branch

On Change option becomes set. If they are based on a label, a promotion state, or a moment in time, they are read-only unless the Branch On Change option is set.

Version control

Version control is the process of storing and tracking the various changes (revisions) to one or more files. A version control system maintains the revision history generated as the files evolve into their final forms.

The main advantage of using an automated version control system is a fast, easy recall of previous revisions.

StarTeam also tracks revisions of change requests, tasks, and topics.

View

Consists of a StarTeam folder hierarchy and the items associated with each folder in that hierarchy either frozen at a specific moment or time or starting from that moment in time and differentiating itself from its parent view afterwards.

A StarTeam project and its root view are identical. Other views can start with a different StarTeam folder as its root, etc.

The behavior of the view may or may not permit branching, may or may not be read only, and may be blank (having no apparent connection to the parent view—an empty subset of the parent view's contents).

View Compare/Merge

You can compare and merge any two views as long as the projects use the same server configuration. For example, as one company started to develop its 3.0 product, the StarTeam administrator created a variant view for that product. The view was derived from the view of the 2.0 version of that same product. However, development continued on the 2.0 product. Two service packs were created and finalized. At that point in time, the development team wanted to merge the source code (text files) from the 2.0 product view with the source code files in the 3.0 product view. StarTeam makes it easy to compare and then merge text files.

View Label

The main purpose of a view label is to "time stamp" the entire contents of the view. Then you can roll back the view to that label and see everything that existed at that point in time. However, unless the view label is frozen, you can make some

adjustments. You can include or exclude some folders and items from the view label by attaching or detaching the view label. You can also move a view label from one revision to another. For example, a couple of files might not have been checked in prior to the creation of the label but need to be included.

View labels are automatically and immediately attached to all folders and items in the view at the time they are created.

Visual Diff

A StarTeam utility that compares two text files or two revisions of one text file and shows the differences (if any) between them.

Web Connect

A StarBase product that allows you to access StarTeam VirtualTeam Server over the Internet.

Working Folder

Every StarTeam folder has a corresponding working folder to which the working copies of files are checked out and from which files are added and checked in to the StarTeam folder.

Working File

A copy of a file revision that has been checked out. StarTeam copies working files to the designated working folder on a workstation.