

Comparison:

Perforce and CVS

Perforce 2009.1 and CVS with New Appendix: 2012 Update – What’s Changed in Perforce

This document compares Perforce (version 2009.1) and CVS (CVSNT version 2.5.04.3510). Read this comparison to:

- See what has been updated in Perforce in 2012
- Understand Perforce and CVS’s major feature differences
- See head-to-head metrics for operations like branching and merging

Table of Contents

Executive Summary	1
Branching and Merging	1
Distributed Development	2
Atomic Transactions	2
Scalability and Performance	2
Performance Tests	3
Test Environment	3
Test Procedure	4
Test Data	4
Test Results	4
Defect Tracking	5
Integrations with Related Tools	5
System Administration and Support	5
Appendix A: 2012 Update – What’s Changed in Perforce	6
Executive Summary	6
Deployment and Administration	6
Distributed and Offline Support	6
Scalability	6
Performance	6
Administration	6
Binary File Management	6
Extensibility	6
Branching and Usage	7
Branching and Release Management	7
Ease of Use	7
Learn More	7
Evaluating Perforce	7
Scheduling a Demo	7
Migrating to Perforce	7

Executive Summary

This document compares Perforce (version 2009.1) with CVS (CVSNT version 2.5.04.3510). Rather than compare every available feature in the two products, it focuses on their most significant differences, such as performance, atomic transactions, and distributed development.

Overview

Attribute	CVS	Perforce
Branching and merging	CVS offers branching, but requires manual tracking of merge history.	Perforce branching automatically tracks the history of all branching operations.
Distributed development	No performance solutions for remote development are available.	The Perforce Proxy offers a caching solution for remote users, with minimal administrative overhead, and at no extra cost.
Atomic transactions	Doesn't have an atomic change mechanism and cannot group changes to related files.	Natively supported as changelists, enabling users to track file versions associated with a feature addition or issue resolution.
Scalability and performance	Metadata is appended and maintained in the content of each versioned file.	Centralized metadata records all user and file activity. Deployed in 10,000+ user environments.
Defect tracking	Not part of CVS; need to use a separate defect tracking solution.	Perforce provides a basic defect tracking system called jobs, and integrates with leading third-party defect tracking systems.
Integration with related tools	Integrations are available via the open source community.	Many third-party tools are designed specifically to work with Perforce.
Support	Available at additional cost from a third party.	Relied upon by over 400,000 users.

Branching and Merging

CVS branching doesn't keep a history of operations, such as integrations between files or branches. Users must track integrations manually. No automated conflict resolution is available when merging files. CVS branches files with the same mechanism it employs to label files. Because all metadata is maintained in the physical versioned file, this branching method requires that each versioned file be opened and updated to include branch information. Locks are required for all targeted files during the branch operation, which can impose long wait times on other participating developers.

The Perforce branching model is powerful and flexible. It's capable of branching thousands of files, and tracks all merges between branches. Merge tracking is an important characteristic for synchronizing branches. Perforce keeps track of all incremental changes as they are introduced to original and branched versions of files. Instead of manually tracking changes across branches, users can rely upon Perforce to integrate file changes across multiple branches automatically. This enables a variety of development scenarios, such as client-specific versions, experimental branches, personal or task branches, and the classic release branching patterns. A built-in graphing tool (see Figure 1) displays the branching history of each file.

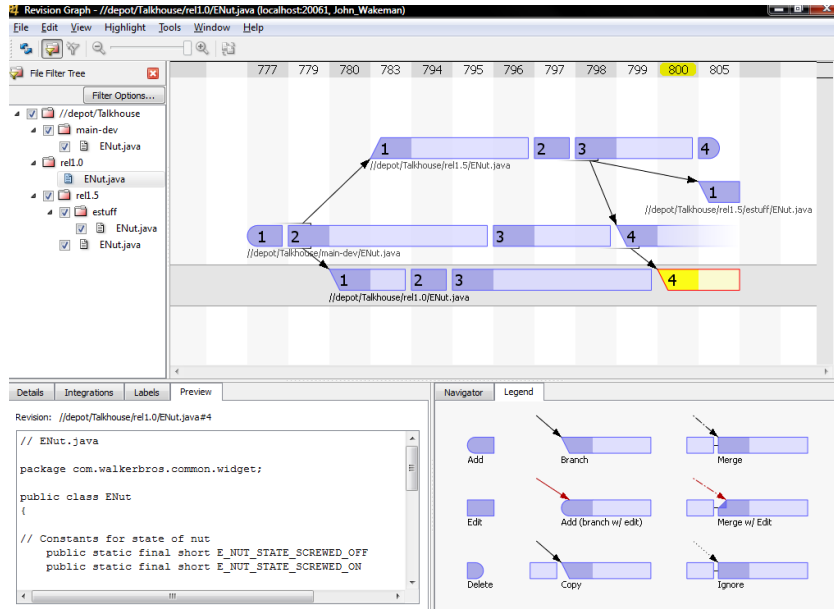


Figure 1: The Perforce Revision Graph shows the branching history of a file, together with its merge history.

Distributed Development

CVS doesn't offer a scalable remote development solution. Users must remotely access a central server. This solution performs when users have a fast, reliable network connection, but there are no efficiencies for groups of users at remote locations. Replication solutions are commercially available from third-party providers.

Perforce's distributed architecture includes fully replicated servers, improved remote depots, and P4Sandbox (see Appendix A).

The Perforce Proxy caches and serves files to users at remote locations, thereby reducing traffic across slower WAN links.

All users, local or remote, connect to the same central depot, and see the exact same project files. Distributed development with Perforce doesn't create any additional process overhead, and the Perforce Proxy requires minimal administrative attention (see Figure 2).

Atomic Transactions

CVS doesn't have an atomic change mechanism and cannot group changes to related files. Relationships between files at check-in are lost, unless artificial methods such as check-in time or common comments are used. If some of the files involved in a check-in operation are rejected, the codebase is left in an inconsistent state.

Perforce organizes the changes made to multiple files into units of work called changelists. Typically, changelists represent features or bug fixes that are implemented by modifying multiple files. Because changelists are atomic, they ensure the integrity of each check-in, and avoid the corruption introduced by partial file submissions. Perforce guarantees that whenever a changelist is submitted, the state of the entire system before and after the changelist varies only by the set of changes (adds, deletes, and edits) involved in the given changelist.

Scalability and Performance

The CVS architecture doesn't support a centralized database for tracking branch, label, and change history. Therefore, all change history must be stored in each versioned file for the duration of its lifetime in RCS. This includes all branch and label information. User updates of client file information require a time-consuming comparison of the client environment to the repository. As the number and size of files increase in the repository, all associated software version management operations take more time to execute. Concurrent users are often placed in a wait state as requested files are updated with branch and label information. Performance for these operations can only be improved by removing stale file history from the physical files.

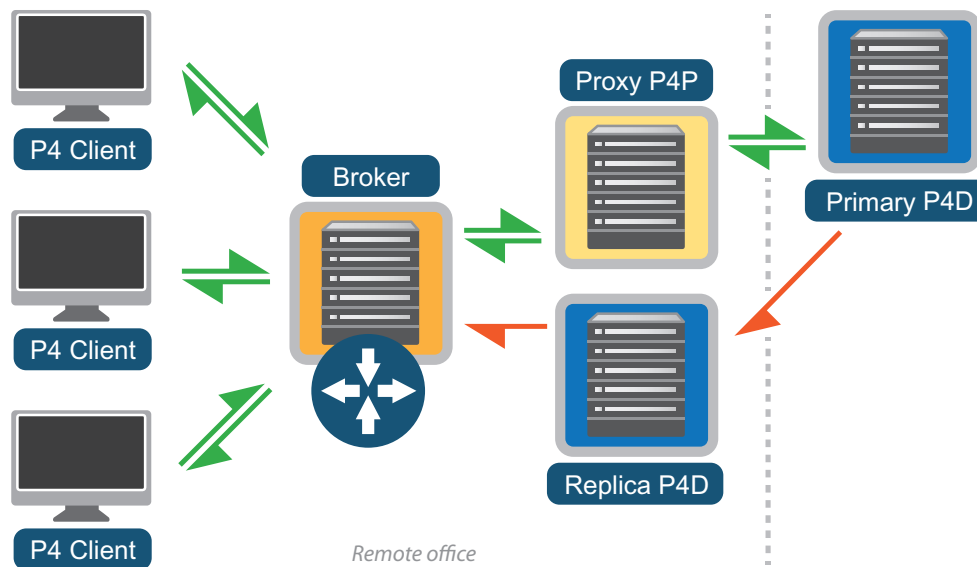


Figure 2: Performce's built-in distributed development architecture.

Performce is architected first and foremost for speed. Performce scales linearly, so there's no performance penalty imposed by the number of revisions of any given file or the size of the file. Performce's built-in relational database maintains all metadata and segregates it from the associated files. Labeling, branching, and reporting are accomplished directly in the database, instead of updates and scans of all the files in a particular branch tree. The Performce Server has been deployed successfully in environments that support more than 10,000 users.

Performance Tests

To test CVS performance against Performce, comparative benchmarking tests were constructed to reflect real-life steps developers would take in carrying out their day-to-day work. In some cases, individual commands were used. In other cases, scripts orchestrating multiple commands were employed. In all cases, functionally equivalent sets of commands were used and executed using the command line utilities.

This comparison provides a look at these two products as they might be used in a real-world environment.

Test Environment

The test environment was selected to provide a common development environment contemporary to 2009. The configuration exceeded the minimum recommendations for both CVS and Performce. The environment consisted of the following:

- Network: Both server and client on same segment of 1 Gbps network
- Server OS: Windows Server 2008 (Standard) SP1 (32 bit)
- Client OS: Windows XP Professional SP2 (64 bit)
- Server Hardware: Dell PowerEdge 1850 with dual 3.3 GHz Xeon processor X5470, 132GB hard disk and 16GB RAM
- Client Hardware: Dell Precision 490 with Intel Xeon 5130 2.0 GHz processor and 2GB RAM

Test Procedure

Testing began with both source control systems in a newly installed state. Tests were run on each system alternately. All developer and source import tests were automated and timed using the DOS time command to a granularity of .01 second. Software installation was timed externally.

Where tests would invoke the file system cache or database cache, the tests were run three times, and average times for second and third iterations were recorded. This simulates an environment where many files are accessed repeatedly through builds and source code searches.

A common file structure was used. Each product created, deleted, and modified the same files during testing. File copies were used to simulate file creation, and automated edits were used to simulate file modifications.

When a sequence of commands was required to complete a single test for either system, a DOS batch file was used to execute the test for both systems.

No performance-oriented techniques were used to optimize the use of either product, other than those used to remove interactive effects.

The tests were designed to compare the two products during normal developer usage. For all processing, automated merges were used, as the tests were constructed to avoid file conflicts. The following tasks were tested:

- Server installation
- Client installation
- Importing source from a local file system
- Populating the workspace
- Updating the workspace (with no changes on the server)
- Labeling all files
- Deleting files on the server
- Undeleting files on the server
- Branching files
- Merging back to the parent branch
- Branching binary files

Test Data

Test data was a combination of two open source software packages: Apache httpd 2.2.14 and Apache Tomcat 6.0.20. A “small” project contained one copy of each package: 4,480 files totaling 56.3MB. A “large” project contained four copies of each package in separate trees: 33,920 files totaling 417MB. Tests for the large project performed the same operations in each tree. Test data for binary files consisted of 217 files (MPG, EXE, and TAR) totalling 1GB.

Test Results

Installing CVS was a larger task than installing Perforce, but not drastically. Perforce outperformed CVS in 16 out of 18 tests. The detailed test results are below. All times are in seconds except where otherwise noted.

Performance Results for Server and Client Installation

Task	CVS	Perforce
Install server	20 mins.	10 mins.
Install client	20 mins.	10 mins.

Performance Results for Small and Large Projects (all times are in seconds except where noted)

Task	Small Project		Large Project	
	CVS	Perforce	CVS	Perforce
Import source	13.78	9.40	51.3	41.10
Populate workspace	34.81	9.97	116.08	80.70
Update workspace (no updates)	6.39	3ms	61.76	39ms
Label/baseline all files	14.27	11ms	47.87	36ms
Delete files on server	58.32	2.0	232.13	11.81
Undelete files on server	33.11	27.10	109.10	136.70
Branch files	14.81	9ms	44.63	2.90
Merge back to parent branch	15.60	49ms	15.82	1.34
Binary files: add files	N/A	N/A	55.19	125.05
Binary files: branch files	N/A	N/A	8.05	8ms

Defect Tracking

CVS doesn't include defect tracking.

Perforce jobs provide an audit trail by enabling bugs and patches to be traced between branches and projects. Perforce can tell you whether or not a bug fix is present in any codeline, regardless of where the fix originated. Jobs are customizable and can be used to track all tasks and link them to work completed, as well as to integrate with third-party defect tracking systems. Data entered into a Perforce job is automatically replicated in the defect tracking tool, and vice versa.

Integrations with Related Tools

The open source community supports CVS. Different GUIs and plug-ins for working within popular IDEs are available.

Perforce has a mature multiplatform GUI, and plug-ins for leading IDEs. Integrations with leading third-party technologies include:

- Defect tracking systems
- Software build and test tools
- Graphical tools

- IDEs
- Merge and diff tools
- Agile tools
- Code review tools

System Administration and Support

CVS has the advantage of full accessibility to the source code, but it's only useful if you're skilled in C programming and can afford the time to delve into it.

Commercial support for CVS is available from various third parties.

Perforce's self-contained software version management system is easy to maintain and frees valuable engineering talent from mounting dedicated disk volumes, configuring file systems, or worrying about third-party licenses.

Expert and responsive technical support is a hallmark of Perforce and full technical support is included during an evaluation.

Appendix A: 2012 Update – What’s Changed in Perforce

This document details changes to Perforce and CVS since the previous comparison paper was published. The original paper compared Perforce version 2009.1 with CVSNT version 2.5.04.3510. This update will consider the latest version of Perforce (version 2011.1).

Although CVSNT 2.8 is available, it is only available as part of a commercial distribution, and cannot be considered part of the generally available CVS distribution.¹

Executive Summary

Since 2009, Perforce has introduced several innovative new features, and expanded its arsenal of tools for managing distributed development and managing complex projects. As a result, the major conclusions of the previous paper, which highlighted Perforce’s advantages in speed, flexibility, and branching, are still valid.

Deployment and Administration

Distributed and Offline Support

Perforce’s support for distributed and offline work has improved tremendously since the 2009.1 release. Perforce’s deployment architecture now includes fully replicated servers, improved remote depots, and P4Sandbox. P4Sandbox is a significant innovation that bridges the gap between distributed and centralized version control. It allows for fully independent private or local work while still maintaining a strong relationship to the central server.

Scalability

Perforce’s deployment architecture is scaling out in two dimensions. First, the fully replicated servers continue to evolve to serve build farms and other automated processes. Second, the replicated servers and P4Sandbox offer improved local and offline performance for users at any location. These tools continue to evolve to shift more work away from the central server.

Performance

Perforce has seen significant performance improvements since the 2009.1 release, including a faster integration engine and improved concurrency support.

Administration

Perforce has continued to evolve its administration tools, with an updated administration GUI, support for coordinated authentication and changelists between several servers, dynamic configuration, and other improvements. The addition of replicated servers helps Perforce scale to support larger user bases, automation, and out-of-the-box disaster recovery with minor administrative overhead. Other Perforce components, such as P4Sandbox, have no extra administrative overhead. Streams actually reduce administrative support requirements in many cases, by incorporating client view management and workflow management best practices into the core product.

Binary File Management

Many industries, including EDA and gaming, must consider a wide variety of digital assets beyond software source code. Graphics, animation clips, story boards, wire frames, and chip designs all comprise important intellectual property and must be versioned in the same manner as text-based assets.

Perforce has improved its support for handling binary files. Large digital assets can be stored in external storage units, and Perforce can be configured on a file-by-file basis to retain only a specified number of revisions of these assets, purging older versions. Obsolete assets can also be archived to offline storage, rather than purged.

Extensibility

Perforce has fully supported APIs for Perl, Ruby, Python, PHP, and Java. The Perforce Broker provides a framework for additional guidance and management of user activity, while the JavaScript API allows users to add visual extensions to Perforce clients. P4toDB allows report engines and other data mining tools to access Perforce metadata through a standard relational database like MySQL or SQL Server.

¹ See <http://www.evscm.org/modules/Downloads/>

Branching and Usage

Branching and Release Management

Perforce's revamped integration engine is substantially faster and now handles the most complex branching and merging scenarios. Non-content changes, such as renames, refactorings, and file type changes, receive special handling.

Perforce Streams provide a lightweight branching and release management framework. Streams help to model a project's branching structure, including how branches relate to one another, the intended merge pathways, and what components are included in a stream. This information is used to simplify and automate common operations such as creating workspaces and merging changes between streams. Visual tools like the Stream Graph provide a rich set of information to all users.

Ease of Use

Perforce introduced shelving, or the ability to save work-in-progress on the server without officially committing it. Shelving is a powerful feature with a variety of uses, such as task switching, collaborating with other users for task handoff, cross-platform pre-submit testing, and code reviews.

Learn More

Evaluating Perforce

More than 400,000 users at 5,500 companies rely on Perforce for enterprise version management. Download a 45-day trial evaluation. Free technical support is included while you evaluate. Get started: perforce.com/trial

Scheduling a Demo

To learn more about Perforce, schedule an interactive demo tailored to your requirements: perforce.com/product/demos

Migrating to Perforce

Perforce Consulting Services has experience assisting customers with migrations from various legacy software version management systems. For more information visit: perforce.com/consulting

perforce.com



North America
Perforce Software Inc.
2320 Blanding Ave
Alameda, CA 94501
USA
Phone: +1 510.864.7400
info@perforce.com

Europe
Perforce Software UK Ltd.
West Forest Gate
Wellington Road
Wokingham
Berkshire RG40 2AT
UK
Phone: +44 (0) 845 345 0116
uk@perforce.com

Australia
Perforce Software Pty. Ltd.
Suite 3, Level 10
221 Miller Street
North Sydney
NSW 2060
AUSTRALIA
Phone: +61 (0)2 8912-4600
au@perforce.com