**Comparison between Version Control Tools**

**-Nihar Raichada (15IT132)**

**1. Apache Subversion**

**Apache Subversion** (often abbreviated **SVN**, after its command name *svn*) is a [software versioning](https://en.wikipedia.org/wiki/Software_versioning) and [revision control](https://en.wikipedia.org/wiki/Revision_control) system distributed as [open source](https://en.wikipedia.org/wiki/Open_source) under the [Apache License](https://en.wikipedia.org/wiki/Apache_License). Software developers use Subversion to maintain current and historical versions of files such as [source code](https://en.wikipedia.org/wiki/Source_code), web pages, and documentation. Its goal is to be a mostly compatible successor to the widely used [Concurrent Versions System](https://en.wikipedia.org/wiki/Concurrent_Versions_System) (CVS).

**Features:**

* [Commits](https://en.wikipedia.org/wiki/Commit_(data_management)) as true [atomic operations](https://en.wikipedia.org/wiki/Atomicity_(database_systems)) (interrupted commit operations in CVS would cause repository inconsistency or corruption).
* Renamed/copied/moved/removed files retain full revision history.
* The system maintains [versioning](https://en.wikipedia.org/wiki/Software_versioning) for directories, renames, and file [metadata](https://en.wikipedia.org/wiki/Metadata) (but not for timestamps). Users can move and/or copy entire directory-trees very quickly, while retaining full revision history.
* Versioning of [symbolic links](https://en.wikipedia.org/wiki/Symbolic_link).
* Native support for binary files, with space-efficient binary-diff storage.
* [Apache HTTP Server](https://en.wikipedia.org/wiki/Apache_HTTP_Server) as network server, [WebDAV](https://en.wikipedia.org/wiki/WebDAV)/[Delta-V](https://en.wikipedia.org/wiki/WebDAV#Extensions_and_derivatives) for [protocol](https://en.wikipedia.org/wiki/Protocol_(computing)). There is also an independent server [process](https://en.wikipedia.org/wiki/Process_(computing)) called svnserve that uses a custom protocol over [TCP/IP](https://en.wikipedia.org/wiki/Internet_Protocol_Suite).
* [Branching](https://en.wikipedia.org/wiki/Branching_(software)) is a cheap operation, independent of file size (though Subversion itself does not distinguish between a branch and a directory)
* Natively [client–server](https://en.wikipedia.org/wiki/Client–server_model), [layered](https://en.wikipedia.org/wiki/Abstraction_layer) [library](https://en.wikipedia.org/wiki/Library_(computing)) design.
* Costs proportional to change size, not to data size.
* Path-based authorization.
* [Language bindings](https://en.wikipedia.org/wiki/Language_binding) for [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [PHP](https://en.wikipedia.org/wiki/PHP), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Perl](https://en.wikipedia.org/wiki/Perl), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), and [Java](https://en.wikipedia.org/wiki/Java_(programming_language)).
* Merge tracking – Merges between branches will be tracked, this allows automatic merging between branches without telling Subversion what does and does not need to be merged.
* Change-lists to organize commits into commit groups.

**2. Git**

### Git is a [version control system](https://en.wikipedia.org/wiki/Version_control_system) for tracking changes in [computer files](https://en.wikipedia.org/wiki/Computer_file) and coordinating work on those files among multiple people. It is primarily used for [source code management](https://en.wikipedia.org/wiki/Source_code_management) in [software development](https://en.wikipedia.org/wiki/Software_development), but it can be used to keep track of changes in any set of files. As a [distributed revision control](https://en.wikipedia.org/wiki/Distributed_revision_control) system it is aimed at speed, data integrity, and support for distributed, non-linear workflows.

### Features

* **Strong support for non-linear development**: Git supports rapid branching and merging, and includes specific tools for visualizing and navigating a non-linear development history. In Git, a core assumption is that a change will be merged more often than it is written, as it is passed around to various reviewers.
* **Distributed development**: Like [Darcs](https://en.wikipedia.org/wiki/Darcs), [BitKeeper](https://en.wikipedia.org/wiki/BitKeeper), [Mercurial](https://en.wikipedia.org/wiki/Mercurial), [SVK](https://en.wikipedia.org/wiki/SVK), [Bazaar](https://en.wikipedia.org/wiki/Bazaar_(software)), and [Monotone](https://en.wikipedia.org/wiki/Monotone_(software)), Git gives each developer a local copy of the full development history and changes are copied from one such repository to another. These changes are imported as added development branches, and can be merged in the same way as a locally developed branch.
* **Compatibility with existent systems and protocols**: Repositories can be published via [Hypertext Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP), [File Transfer Protocol](https://en.wikipedia.org/wiki/File_Transfer_Protocol) (FTP), [rsync](https://en.wikipedia.org/wiki/Rsync) (removed in Git 2.8.0), or a Git protocol over either a plain socket, or [Secure Shell](https://en.wikipedia.org/wiki/Secure_Shell) (ssh). [Subversion](https://en.wikipedia.org/wiki/Apache_Subversion) and [svk](https://en.wikipedia.org/wiki/Svk) repositories can be used directly with git-svn.
* **Efficient handling of large projects:** Torvalds has described Git as being very fast and scalable and performance tests done by Mozilla showed it was an [order of magnitude](https://en.wikipedia.org/wiki/Order_of_magnitude) faster than some version control systems.
* **Cryptographic authentication of history:** The Git history is stored in such a way that the ID of a particular version (a *commit* in Git terms) depends upon the complete development history leading up to that commit. Once it is published, it is not possible to change the old versions without it being noticed.
* **Toolkit-based design:** Git was designed as a set of programs written in [C](https://en.wikipedia.org/wiki/C_(programming_language)), and several shell scripts that provide wrappers around those programs. Although most of those scripts have since been rewritten in C for speed and portability, the design remains, and it is easy to chain the components together.
* **Pluggable merge strategies:** As part of its toolkit design, Git has a well-defined model of an incomplete merge, and it has multiple algorithms for completing it, culminating in telling the user that it is unable to complete the merge automatically and that manual editing is needed.
* [**Garbage**](https://en.wikipedia.org/wiki/Garbage_(computer_science)) **accumulates until collected:** Aborting operations or backing out changes will leave useless dangling objects in the database. Git will automatically perform [garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) when enough loose objects have been created in the repository. Garbage collection can be called explicitly using git gc --prune.

**3. GNU Bazaar**

**GNU Bazaar** (formerly **Bazaar-NG**, command line tool **bzr**) is a [distributed](https://en.wikipedia.org/wiki/Distributed_version_control) and [client–server](https://en.wikipedia.org/wiki/Client–server) [revision control](https://en.wikipedia.org/wiki/Revision_control) system sponsored by Canonical.

**Features:**

* Bazaar can be used by a single developer working on multiple branches of local content, or by teams collaborating across a network.
* Bazaar is written in the [Python programming language](https://en.wikipedia.org/wiki/Python_programming_language), with packages for major [GNU/Linux](https://en.wikipedia.org/wiki/Linux) distributions, [Mac OS X](https://en.wikipedia.org/wiki/Mac_OS_X) and [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows). Bazaar is free software and part of the [GNU Project](https://en.wikipedia.org/wiki/GNU_Project).[[4]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-febgnu-4)[[5]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-5)
* Bazaar commands are similar to those found in [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System) or [Subversion](https://en.wikipedia.org/wiki/Subversion_(software)). A new project can be started and maintained without a remote repository server by invoking bzr init in a directory which a person wishes to version.[[6]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-bzr-man-page-6)
* In contrast to purely distributed version control systems which do not use a central server, Bazaar supports working with or without a central server.[*[clarification needed](https://en.wikipedia.org/wiki/Wikipedia:Please_clarify)*] It is possible to use both methods at the same time with the same project. The websites [Launchpad](https://en.wikipedia.org/wiki/Launchpad_(website)) and [Sourceforge](https://en.wikipedia.org/wiki/Sourceforge) provide free hosting service for projects managed with Bazaar.
* Bazaar has support for working with some other revision control systems.[[7]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-7) This allows users to branch from another system (such as [Subversion](https://en.wikipedia.org/wiki/Subversion_(software))[[8]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-8)), make local changes and commit them into a Bazaar branch, and then later merge them back into the other system. Read-only access is also available for [Git](https://en.wikipedia.org/wiki/Git_(software))[[9]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-9) and [Mercurial](https://en.wikipedia.org/wiki/Mercurial_(software)).[[10]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-10) Bazaar also allows for interoperation with many other systems (including [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System), [Darcs](https://en.wikipedia.org/wiki/Darcs), [Git](https://en.wikipedia.org/wiki/Git_(software)), [Perforce](https://en.wikipedia.org/wiki/Perforce), [Mercurial](https://en.wikipedia.org/wiki/Mercurial)) by allowing one to import/export the history.[[11]](https://en.wikipedia.org/wiki/GNU_Bazaar#cite_note-11)
* Bazaar supports files with names from the complete [Unicode](https://en.wikipedia.org/wiki/Unicode) set. It also allows commit messages, committer names, etc. to be in Unicode.

**Git Vs Subversion**

* Git is much faster than Subversion
* Subversion allows you to check out just a subtree of a repository; Git requires you to clone the entire repository (including history) and create a working copy that mirrors at least a subset of the items under version control.
* Git's repositories are much smaller than Subversions (for the Mozilla project, 30x smaller)
* Git was designed to be fully distributed from the start, allowing each developer to have full local control
* Git branches are simpler and less resource heavy than Subversion's
* Git branches carry their entire history
* Git provides better auditing of branch and merge events
* Git's repo file formats are simple, so repair is easy and corruption is rare.
* Backing up Subversion repositories centrally is potentially simpler - since you can choose to distributed folders within a repo in git
* Git repository clones act as full repository backups
* Subversion's UI is more mature than Git's

# **Git Vs Bazaar**

## **Bazaar has better Windows Support:** Git currently has beta-level native Windows support, code-named [MSysGit](http://code.google.com/p/msysgit). Bazaar, on the other hand, includes a native Windows port and installer.

## **Bazaar has less attitude — direct support for more workflows as compared to Git:** Bazaar's UI directly supports a larger range of work flows than Git.

## **Bazaar has easier administration facilities than Git:** Bazaar and Git make creating repositories trivial, but the ongoing cost of ownership shouldn't be ignored. Both Bazaar and Git have tools for detecting junk and inconsistencies in repositories. Bazaar also has an upgrade tool for switching between file formats and a reconfigure tool for changing how a branch is configured, e.g. lightweight vs heavyweight, standalone vs shared repository.

## **Robust renaming in Bazaar as compared to Git:** Git prides itself on being a content manager and deriving what got renamed using heuristics. This mostly works, but breaks under certain merge conditions. If you want your team or community to collaborate without fear of breaking merges, Bazaar's robust renaming is essential.

## **Bazaar has better asynchronous sharing:** When changes in a branch are ready for sharing and you wish to share asynchronously (e.g. via email instead of advertising a public branch), Bazaar handles this better than Git.

## **Git has better speed as compared to Bazaar:** We want Bazaar to be the most usable tool around and performance is an important part, but certainly not the only part, of achieving that. It's true that Git is really fast at many operations and that Bazaar was once quite slow.

## **Storage efficiency is better in Git:** Git has a strong reputation for efficiently storing data and claims on its home page that it tops every other open source VCS in this area. Since Bazaar changed its default format to packs in 1.0, studies indicate that Bazaar is around 15% better on average for the initial commit.

## **Cryptographic content validation in Git:** Linus made cryptographic strength integrity checking a core part of Git’s design. As revisions are named using their SHA, it's next to impossible to attack a Git repository.

**Bazaar Vs Subversion**

* SVN **requires** all commits to go to a central location, and tends to favor having multiple people working on the same branch. Bazaar has chosen to address this with workflows. However, with a Bazaar checkout, there is always the possibility to either bzr unbind or just bzr commit --local when you are on a plane, or just want to record in-progress work before integrating it into the master branch.
* SVN has a lot more 3rd party support.
* Bazaar is generally much easier to set up. SVN can only really be set up by an administrator. Getting a project using Bazaar is usually as simple as bzr init; bzr add; bzr commit -m initial import.
* Subversion's model is a giant versioned filesystem. Bazaar uses a concept of a Tree.
* SVN supports partial checkouts better than Bazaar does.
* Rename support in SVN is much better than Bazaar.
* Bazaar supports better merging - fully integrated and works across distributed repositories