Version control

Version control systems are a category of software tools that help a software team manage changes to source code over time. Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

For almost all software projects, the source code is like the crown jewels - a precious asset whose value must be protected. For most software teams, the source code is a repository of the invaluable knowledge and understanding about the problem domain that the developers have collected and refined through careful effort. Version control protects source code from both catastrophe and the casual degradation of human error and unintended consequences.

Software developers working in teams are continually writing new source code and changing existing source code. The code for a project, app or software component is typically organized in a folder structure or "file tree". One developer on the team may be working on a new feature while another developer fixes an unrelated bug by changing code, each developer may make their changes in several parts of the file tree.

Version control helps teams solve these kinds of problems, tracking every individual change by each contributor and helping prevent concurrent work from conflicting. Changes made in one part of the software can be incompatible with those made by another developer working at the same time. This problem should be discovered and solved in an orderly manner without blocking the work of the rest of the team. Further, in all software development, any change can introduce new bugs on its own and new software can't be trusted until it's tested. So testing and development proceed together until a new version is ready.

Good version control software supports a developer's preferred workflow without imposing one particular way of working. Ideally it also works on any platform, rather than dictate what operating system or tool chain developers must use. Great version control systems facilitate a smooth and continuous flow of changes to the code rather than the frustrating and clumsy mechanism of file locking - giving the green light to one developer at the expense of blocking the progress of others.

Software teams that do not use any form of version control often run into problems like not knowing which changes that have been made are available to users or the creation of incompatible changes between two unrelated pieces of work that must then be painstakingly untangled and reworked. If you're a developer who has never used version control you may have added versions to your files, perhaps with suffixes like "final" or "latest" and then had to later deal with a new final version. Perhaps you've commented out code blocks because you want to disable certain functionality without deleting the code, fearing that there may be a use for it later. Version control is a way out of these problems.

Version control software is an essential part of the every-day of the modern software team's professional practices. Individual software developers who are accustomed to working with a capable version control system in their teams typically recognize the incredible value version control also gives them even on small solo projects. Once accustomed to the powerful benefits of version control systems, many developers wouldn't consider working without it even for non-software projects.

As a system admin, the chances are you collaborate with multiple people across the company, therefore you will probably know the stress of constantly transferring files and version controlling the changes. Version control tools are a great way to enable collaboration, maintain versions, and track changes across the team.

Perhaps the greatest benefit of using version control tools is that you have the capacity to deal with an unlimited number of people, working on the same code base, without having to make sure that files are delivered back and forth. Below are some of the most popular and most preferred open-source version control systems and tools available for making your setup easier.

1. **GIT**

Git is considered to be a newer, and faster emerging star when it comes to version control systems. First developed by the creator of Linux kernel, Linus Torvalds, Git has begun to take the community for web development and system administration by storm, offering a largely different form of control. Here, there is no singular centralized code base that the code can be pulled from, and different branches are responsible for hosting different areas of the code. Other version control systems, such as CVS and SVN, use a centralized control, so that only one master copy of software is used.

Advantage -

- As a fast and efficient system, many system administrators and open-source projects use Git to power their repositories. However it is worth noting that Git is not as easy to learn as SVN or CVS is, which means that beginners may need to steer clear if they’re not willing to invest time to learn the tool.
- The benefits of splitting it up into multiple repositories is that there is no longer one single point of failure. With **GIT** as each developer has the own repository it doesn't matter if the master repository is broke they can continue to commit code locally, until the master repository is fixed and then they can push their code into the master repository.

2. SVN

SVN, or Subversion as it is sometimes called, is generally the version control system that has the widest adoption. Most forms of open-source projects will use Subversion because many other large products such as Ruby, Python Apache, and more use it too. Google Code even uses SVN as a way of exclusively distributing code.

Because it is so popular, many different clients for Subversion are available. If you use Windows, then Tortoise SVN may be a great browser for editing, viewing and modifying Subversion code bases. If you're using a MAC, however, then Versions could be your ideal client.

Advantage-

- CVS only tracks modification on a file-by-file basis, while SVN tracks a whole commit as a new revision, which means that it is easier to follow the history of your project.
- it's faster than CSV.
- supports versioning of binary files
- adds transactional commit (all or nothing).

3. Mercurial

This is yet another form of version control system, similar to Git. It was designed initially as a source for larger development programs, often outside of the scope of most system admins, independent web developers and designers. However, this doesn't mean that smaller teams and individuals can't use it. Mercurial is a very fast and efficient application. The creators designed the software with performance as the core feature.

Aside from being very scalable, and incredibly fast, Mercurial is a far simpler system to use than things such as Git, which one of the reasons why certain system admins and developers use it. There aren't quite many things to learn, and the functions are less complicated, and more comparable to other CVS systems. Mercurial also comes alongside a web-interface and various extensive documentation that can help you to understand it better.

Advantage -

- Mercurial's CLI is full-featured, stable, and elegant.
- Mercurial's simple approach leads to documentation which is sleek and concise.
- It's easy to find what you're looking for — using *hg* help.
- Mercurial tries to be helpful by anticipating some of the more common aliases for commands and just making them work. For example, the hg rename command can also be run with hg move or hg mv.

\

4. **CVS**

Concurrent Versions System (CVS) is a program that lets a code developer save and retrieve different development versions of source code . It also lets a team of developers share control of different versions of files in a common repository of files. This kind of program is sometimes known as a *version control system* . CVS was created in the UNIX operating system environment and is available in both Free Software Foundation and commercial versions. It is a popular tool for programmers working on Linux and other UNIX-based systems.

CVS works not by keeping track of multiple copies of source code files, but by maintaining a single copy and a record of all the changes. When a developer specifies a particular version, CVS can reconstruct that version from the recorded changes. CVS is typically used to keep track of each developer's work individually in a separate working directory. When desired, the work of a team of developers can be merged in a common repository. Changes from individual team members can be added to the repository through a "commit" command.

5. **Bazaar**

Similar to Git and Mercurial, Bazaar is distributed version control system, which also provides a great, friendly user experience. Bazaar is unique that it can be deployed either with a central code base or as a distributed code base. It is the most versatile version control system that supports various different forms of workflow, from centralized to decentralized, and with a number of different variations acknowledged throughout. . One of the greatest features of Bazaar is that you can access a very detailed level of control in its setup. Bazaar can be used to fit in with almost any scenario and this is incredibly useful for most projects and admins because it is so easy to adapt and deal with. It can also be easily embedded into projects that already exist. At the same time, Bazaar boasts a large community that helps with the maintenance of third-party tools and plugins.