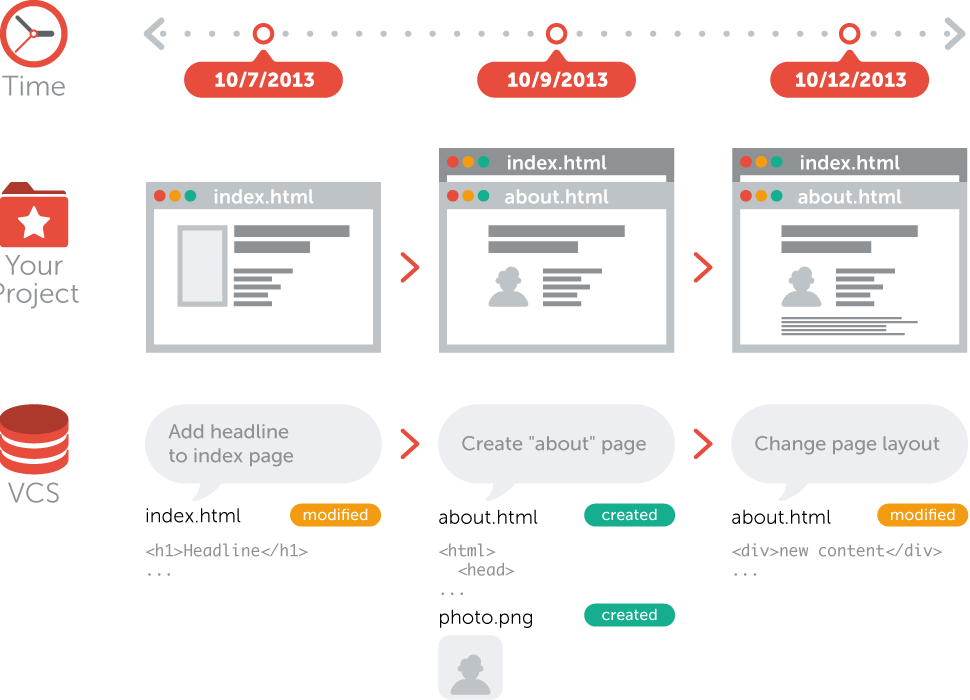
**Alternatives to Git**

**What is Version Control System (VCS)**

VCS is a kind of database that lets you save a snapshot of your complete project and later when you look at the older snapshot, VCS shows how different your current snapshot is from the older one.

****

VCS is not a deployment system where you don’t have to change or replace any other part of your toolchain when you start using version control.

A version control system records the changes you make to your project's files. This is what version control is about. It's as simple as it sounds.

**Git**

Git is one of the most famous and secured version-control system created by Linus Torvalds in 2005 primarily developed on Linux which supports other OS including macOS, Windows, BSD, and Solaris.



### All the repositories of Git are equal which makes it one of the best version control systems out there. It has a web-based hosting service called GitHub which is globally popular with developers uploading their code on GitHub now and then to showcase their credibility on any particular technology. Even employers look up to Github as a virtual resume highlighting the potential of a person.

### Git's popularity in the development community is largely down to the fact that the content could be stored and tracked at regular intervals. It allows to periodically chance the stored code as new codes are added and also maintain a history of each change that is passing through.

Git's level of tracking is unique as unlike other version control systems, branches and tags are also stored in history.

But there are other alternatives to Git which is being used by many companies as the version control system to keep their project safe on cloud repository.

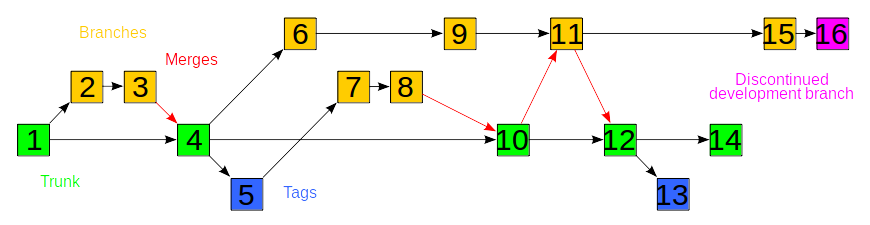
**SVN**



Apache Subversion aka SVN is a software versioning and revision control system distributed as open source under the license of Apache. Subversion project was founded by CollabNet in 2000 to create an open source system based on CVS (Concurrent Version System) with fixed bugs and some new features.

* **Branching and tagging in SVN**

The inter-file branching model is used by Subversion for the implementation of braches which deals with a separate line of development and tagging which adds labels to a repository to be tracked in the future.



* **Subversion Architecture**

**Directory Structure**

Below is a reference to how an SVN project with more than one feature would look like: -

test\_project/path/README.md

test\_project/path/lib/widget.rb

test\_project/path/new\_feature/README.md

test\_project/branches/new\_feature/lib/widget.rb

test\_project/branches/another\_new\_feature/README.md

test\_project/branches/another\_new\_feature/lib/widget.rb

SVN has 3 main advantages:

• SVN is faster than others.

• It supports versioning of binary files

• It adds transactional commit (all or nothing)

**Concurrent Version System(CVS)**

****

CVS is a version control system used worldwide, and with the help of which one could capture the history of source files and documents. It's used in many production software projects as well.

CVS works under the RCS topology which stores files and handles multiple revisions of a single file.

## How does it work?

In CVS, a central system keeps track of the changes committed and avoids conflicts between the server and the client versions of the data. In case of any disputes, however, the changes are proposed and adjusted accordingly.

**Perforce**

****

Perforce is one the most potent version control systems which manage multiple people while working on the same files and it keeps track of the changes to ensure reverting to the prior versions when deemed necessary.

The client-server architecture of Perforce allows one to make changes on a local disk and then submit while the other could update the repository with their files. Though the data are shared through a central repository, a failed submit would be rolled back to its previous state and could be submitted fresh again.

**Advantages of Perforce:**

* Perforce merging is better than Subversion or Git.
* Perforce has excellent customer support.
* Perforce allows combinations of Subversion's repository revisions with individual file revisions.
* In case of multiple change lists, however, Perforce is better than SubVersion as there is no involuntary commitment when a file is moved from a default change list.
* One of the other advantages of Perforce is that the layout of the working directory could be specified.

**Mercurial**

****

Mercurial’s CLI is a full-featured, stable, and elegant version control system which follows the philosophy of individual commands performing a single command with precision. Mercurial’s simple approach leads to sleek and concise documentation. It uses clarity in its architecture which makes it easy to use.

Mercurial also eases the transition between tools much like Subversion. It helps in creating aliases and make it a pleasant interface to use.

**Some of its advantages**:

1) Reliance on the server backup is minimized as the entire copy of the repository could be found on the local system.

2) It works both online and offline which means even without the internet connection, the work could be continued and changes could be committed to the local repository.

3) The organization of the changesets for Mercurial version control system is done via revisions that allow the branching or merging of the code which is difficult in SVN.

**Trends**

Some of the significant changes in Software Release Management which should be adopted by a company.

* The first trend is the adoption of agile practices at the core, including automation which would allow sprint and Iteration planning as well retrospection of all data for successful release. Moreover, companies should invest time and money in automation to ensure relevancy in the stories and features of the sprints.
* These days teams need to test their products in multiple environments before the release which glorifies the importance of cloud platforms in the production environments. Most of the products today starting from operating systems to Internet browsers are supportive of different platforms which make testing essential in all the situations.
* The popularity of distributed version control systems like GIT, Mercurial, etc. resulting from the flexibility it provides for teams to work on different levels make it easier for individuals to keep a self-contained repository on one's system. A developer could work offline without the necessity of a privileged master repository.