Git is a free and open-source version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is a distributed version control system for **tracking changes in source code** during software development. It is designed for coordinating work among programmers, but it can be used to track change.

It allows you to:

* Revert files to previous state,
* Revert entire project back to previous state,
* Compare changes over time,
* See who modified what? **And much more...**

It means if you screw things up or lose files, you can easily recover.

Use cases:

* Individual development,
* Collaborative development,
* Offline usage.

Why Git?

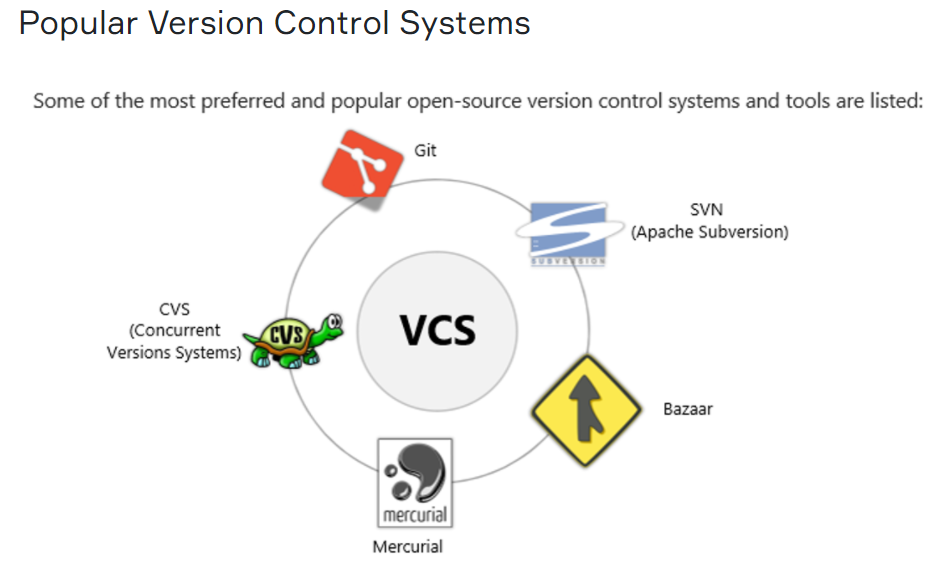
* Everything is local (full history tree available offline),
* Everything is fast,
* Snapshots, not diffs,
* It is distributed not centralized,
* Great for those who hate: CVS/SVN (earlier [version control systems](https://lms.clarusway.com/mod/lesson/view.php?id=642)).

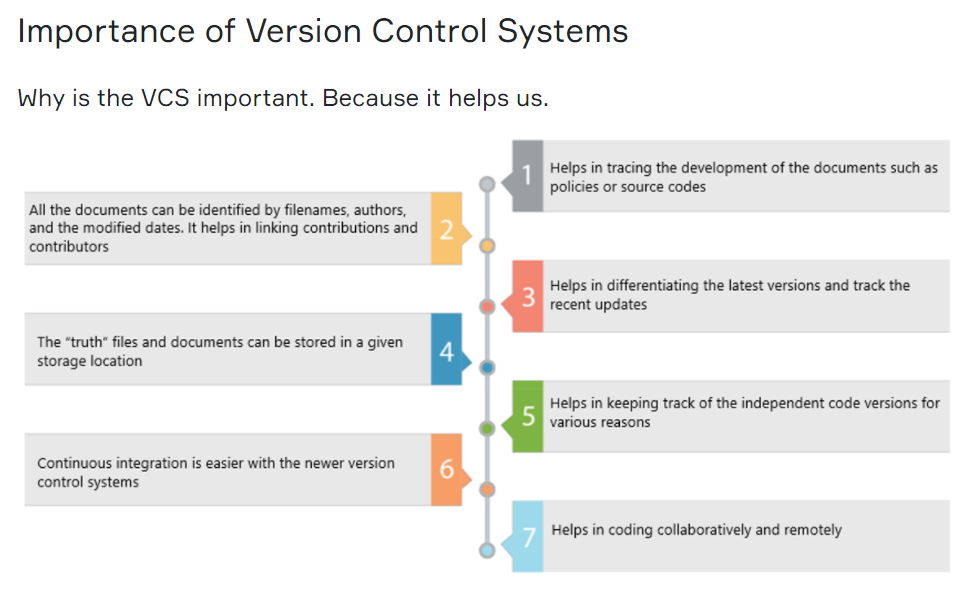
What is a Version Control System (VCS)?

A **V**ersion **C**ontrol **S**ystem (also called Revision/Source Control System) is a software designed to record changes made to files over time so that you can recall specific versions later. It's like having an unlimited undo/redo feature under your belt.  It allows you to revert selected files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more. It's a very powerful tool for your project.

A VCS is a tool to track changes of source code and learning

* What had changed
* When it changed
* Why it changed
* Who changed it



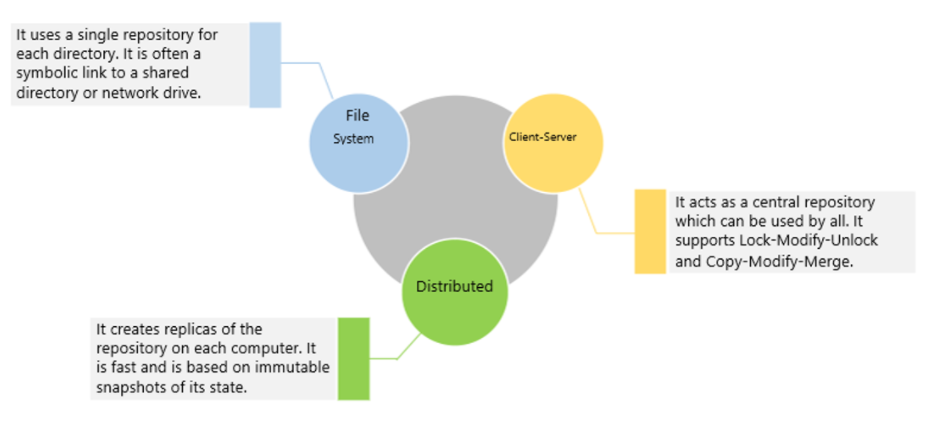


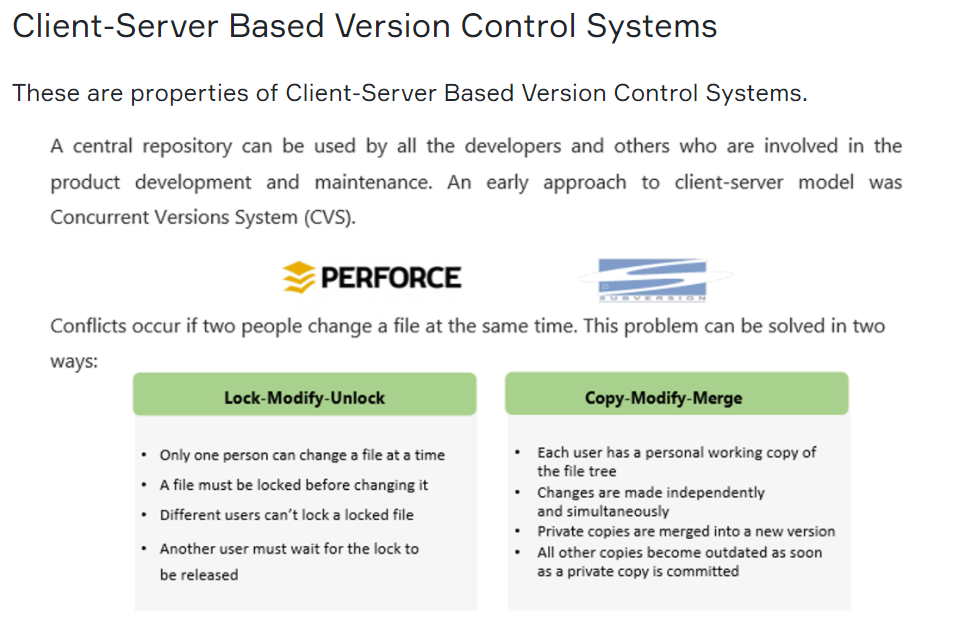
Types of Version Control Systems

There are three types of version control systems:

* File-based,
* Client-server type,
* Distributed.

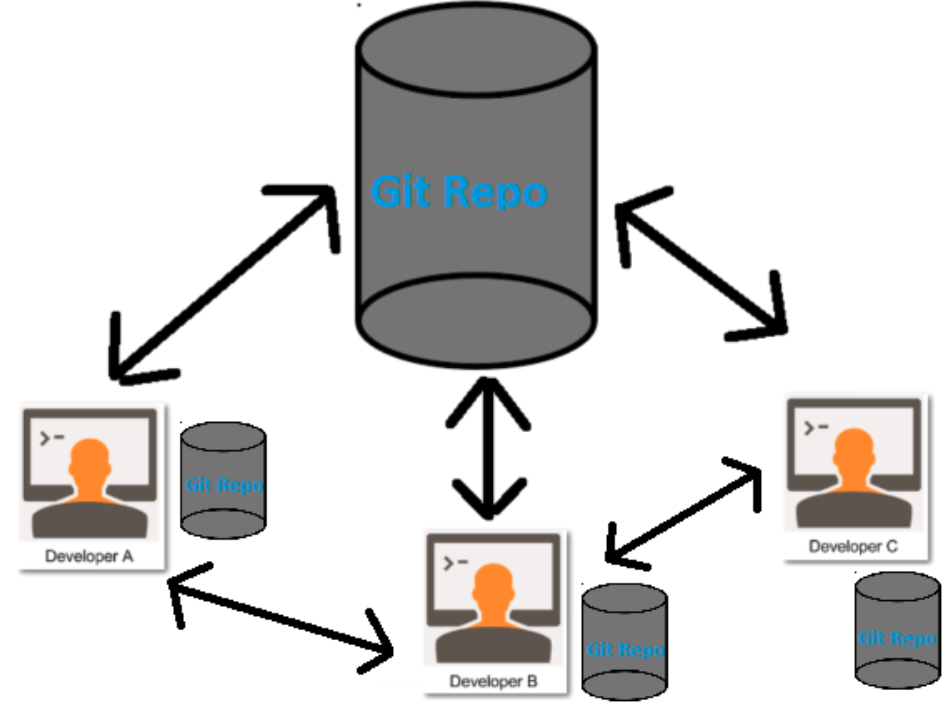
File-based version control systems are obsolete and not much used. Distributed VCS's are more common on the market. Git is a distributed type of VCS.

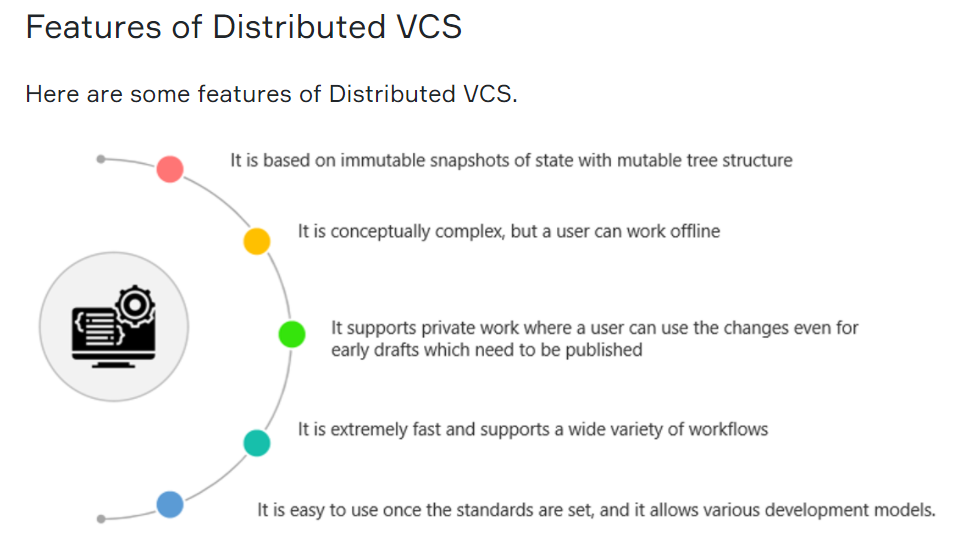




Distributed Version Control Systems

Distributed version control systems create replicas of the repository on each computer. Every user has to work on a replica and can do so even being disconnected from the network. They are suited for large projects and independent developers who can work independently and commit the changes for merging.





**Linux:**

For Ubuntu: $ sudo apt-get install git

For Fedora (works also on **AWS Linux 2** Free Tier AMI): $ sudo yum install git

Once installed, check Git Version: $ git --version

If you see any version number, you are good to go! 👍

You can set your username and email as follows :

$ git config --global user.name "Your Name"

$ git config --global user.email "youremail@example.com"

Enable color in git: $ git config --global color.ui auto

Checking your settings $ git config --list

(press **q** if you need to exit from this list)

### **What is a Repository?**

A repository is a directory or storage space where your projects can live. Sometimes it is shortened to “repo.” It can be local to a folder on your computer, or it can be a storage space on the cloud. You can keep code files, text files, image files, etc. inside a repository.

Let's create a local repo and see what it looks like.

Go to your desktop and create a folder named "git-projects". $ mkdir git-projects

Get into the folder. $ cd git-projects

Make another folder called "project1". $ mkdir project1

And check what is inside the "git-projects" folder. $ ls -al

To create a repo, type $ git init

And to see the updated files, type $ ls -al

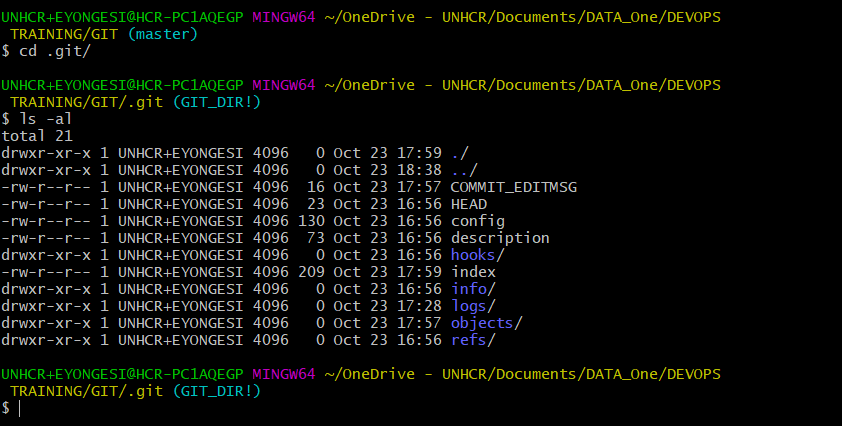
Now you have a repo named .git. It's a secret folder consists of many files and folders.

Let's check what is inside the repo.

$ cd .git

$ ls -al

And this is how a **Repo** looks like. Git understands all the versions of the files and folders in your project and saves them in a special way. All your files and folders will be stored in Git Database.



### **Understanding How Git Manages Data**

In a Git repository your file can reside in three main states: **Modified**, **Staged**, and **Committed**.

* **Modified** means that you have changed the file but have not committed it to your database (repo) yet.
* **Staged** means that you have marked a modified file in its current version to go into your next commit snapshot.
* **Committed** means that the data is safely stored in your local database.

This leads us to the three main sections of a Git project:

* The working tree,
* The staging area,
* The Git directory.

The working tree is a single checkout of one version of the project. These files are pulled out of the compressed database in the Git directory and placed on disk for you to use or modify.

The staging area is a file, generally contained in your Git directory, that stores information about what will go into your next commit. Its technical name in Git terminology is the “index”, but the phrase “staging area” works just as well.

The Git directory (.git) is where Git stores the metadata and object database for your project.