

Git is the most popular tool among all the DVCS tools.

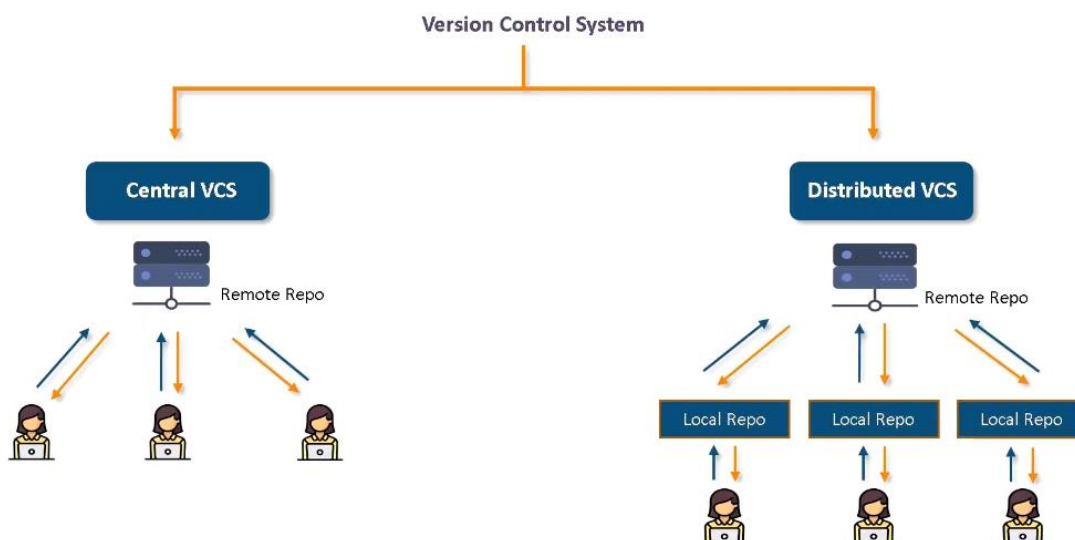
Git is a version-control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source-code management in software development, but it can be used to keep track of changes in any set of files.

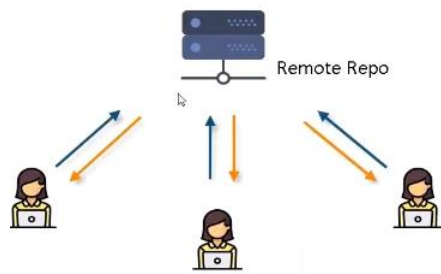
Version control is a system that records/manages changes to documents, computer programs etc over time. It helps us tracking changes when multiple people work on the same project

GIT Features:

- ✓ Versioning is Automatic
- ✓ Team Collaboration is simple
- ✓ Easy Access to previous Versions
- ✓ Only modified code is stored across different versions, hence saves storage

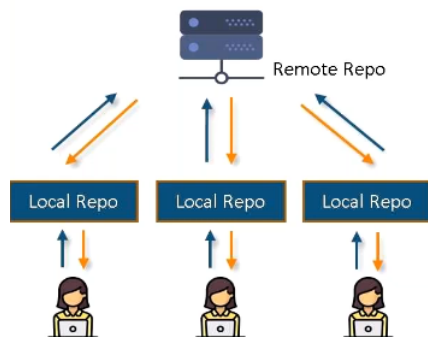
TYPES OF VCS





Centralized VCS

- ★ Centralized Version Control System has one single copy of code in the central server
- ★ Developers will have to “commit” their changes in the code to this central server
- ★ “Committing” a change simply means recording the change in the central system



Distributed VCS

- ★ In Distributed VCS, one does not necessarily rely on a central server to store all the versions of a project's file
- ★ Every developer “clones” a copy of the main repository on their local system
- ★ This also copies, all the past versions of the code on the local system too
- ★ Therefore, the developer need not be connected to the internet to work on the code

Distributed VCS

- ★ Everything except pushing and pulling can be done without Internet Connection
- ★ Every Developer has full version history on local hard drive
- ★ Committing and retrieving action is faster since data is on local drive
- ★ Not Good for storing large files which are binary in nature, this would increase the repo size at every commit
- ★ If a project has a lot of commits, downloading them may take a lot of time

Centralized VCS

- ★ Needs a dedicated internet connection for every operation
- ★ Developers just have the working copy and no version history on their local drive
- ★ Committing and retrieving action is slower since it happens on the internet
- ★ Good for storing large files, since version history is not downloaded
- ★ Not dependent on the number of commits

GIT LIFE CYCLE:

Working Directory

Staging Area

Local Repo

- ★ The place where your project resides in your local disk
- ★ This project may or may not be tracked by git
- ★ In either case, the directory is called the working directory
- ★ The project can be tracked by git, by using the command `git init`
- ★ By doing `git init`, it automatically creates a hidden `.git` folder

Working Directory

Staging Area

Local Repo

- ★ Once we are in the working directory, we have to specify which files are to be tracked by git
- ★ We do not specify all files to be tracked in git, because some files could be temporary data which is being generated while execution
- ★ To add files in the staging area, we use the command `git add`

Working Directory

Staging Area

Local Repository

- ★ Once the files are selected and are ready in the staging area, they can now be saved in repository
- ★ Saving a file in the repository of git is known as doing a commit
- ★ When we commit a repository in git, the commit is identified by a commit id
- ★ The command for initializing this process is `git commit -m "message"`

Local Operations

