SOFTWARE ENGINEERING ASSIGNMENT-3

Submitted by
A. Aparna Menon
15IT152

Version Control Tools

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.

There are different version control tools available for a developer's use. Three of such version control tools are described below:

1.**CVS**:

- CVS stands for 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.
- 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.
- CVS uses another program, Revision Control System (RCS), to do the actual revision management - that is, keeping the record of changes that goes with each source code file.

2. **Tortoise SVN:**

- SVN stands for Subversion.
- Subversion is a centralized system for sharing information.
- At its core is a repository, which is a central store of data. The repository stores information in the form of a filesystem tree - a typical hierarchy of files and directories.
- Any number of clients connect to the repository, and then read or write to these files. By writing data, a client makes the information available to others; by reading data, the client receives information from others.
- Each repository is like a file server.
- Commits are the saved state of code changes for specific points in time. A commit is literally saving your development progress; every commit is a mile marker on your product's roadmap.
- Commits in SVN are done between the local checkout and central repository. Changes are committed to the central repo. Each commit includes both the changes and a commit message, which gives details on the changes you're introducing.
- It is a general system that can be used to manage *any* collection of files. For you, those files might be source code—for others, anything from grocery shopping lists to digital video mixdowns and beyond.

3. **Git:**

- Git is the most commonly used version control system today and is quickly becoming the standard for version control.
- Git is a distributed version control system, meaning your local copy of code is a complete version control repository. These fully-functional local repositories make it is easy to work offline or remotely.
- You commit your work locally, and then sync your copy of the repository with the copy on the server.
- This paradigm differs from centralized version control where clients must synchronize code with a server before creating new versions of code.
- Every time you save your work, Git creates a commit. A commit is a snapshot of all your files at a point in time. If a file has not changed from one commit to the next, Git uses the previously stored file. This design differs from other systems which store an initial version of a file and keep a record of deltas over time.
- Commits create links to other commits, forming a graph of your development history . You can revert your code to a previous commit, inspect how files changed from one commit to the next, and review information such as where and when changes were made.

- Commits are identified in Git by a unique cryptographic hash of the contents of the commit. Because everything is hashed, it is impossible to make changes, lose information, or corrupt files without Git detecting it.
- Branches, which are lightweight pointers to work in progress, manage the separation between different commits. Once your work created in a branch is finished, merge it back into your team's main (or master) branch.

Version Control Tool being used in the project:

The version control tool that will be used in this project is **Git**. This tool has been chosen due to the following advantages of Git over other tools:

1. Simultaneous development: Everyone has their own local copy of code and can works simultaneously on their own branches. Git works when you're offline since almost every operation is local.
2. Feature Branch Workflow**:** One of the biggest advantages of Git is its branching capabilities. Unlike centralized version control systems, Git branches are easy to merge. This facilitates the feature branch workflow popular with many Git users.
3. Pull Requests: Many source code management tools such as GitHub enhance core Git functionality with pull requests. A pull request is a way to ask another developer to merge one of your branches into their repository.