Configuration management

-Meherzad(201112025)

We will be discussing about git as a configuration management tool. It is a widely used CMS and is also gaining popularity among users. SCM systems are based on a simple idea: the definitive copies of your files are kept in a central repository. People check out copies of files from the repository, work on those copies, and then check them back in when they are finished. SCM systems manage and track revisions by multiple people against a single master set.
All SCM systems provide the following essential features:

- Concurrency Management

  Git is distributed type of versioning system so each user has to local copy and also a central copy of code.

- Versioning

  Git thinks of its data more like a set of snapshots of a mini filesystem. Every time you commit, or save the state of your project in Git, it basically takes a picture of what all your files look like at that moment and stores a reference to that snapshot. To be efficient, if files have not changed, Git doesn't store the file again—just a link to the previous identical file it has already stored.

- Synchronization

  Only when the user pushes the data it will go to main repository otherwise the changes are made in the local repository only. And also user can create branches on the central main repository so that the user can merge that to main branch when he is satisfied the branch will not trouble the main code.

Git provides all these features and in best possible way in comparison to other SCM.

Learning git is easy with some quick learning of easy commands user can start using git and now also a GUI tool is available which helps to make the work very much easy in comparison to a command line tool. Initially user has to understand the working of git that how data is stored and how versioning is maintained. There is large community of developers so support is excellent.

GIT is a distributed SCM system. With Subversion, for each project there is a single repository at some detached central place where all the history is and which we checkout and commit into. GIT works differently, each copy of the project tree (we call that the working copy) carries its own repository around (in the .git subdirectory in the project trees root). So we can have local and remote branches. We can also have a so-called bare repository which is not attached to a working copy — that is useful especially when we want to publish our repository.

Each commit has an author and a committer field, which record who and when created the change and who committed it (GIT is designed to work well with patches coming by mail — in that case, the author and the committer will be different).

One thing that is very annoying with SVN is that is stores its metadata all over the place. GIT on the other hand stores it in a single .git folder at the root of the working copy. Everything is there, we do not have .git folders all over the place like with SVN and its .svn folders.

IT has a very strong and huge community, thus development and support is excellent whether one tries the IRC (Internet Relay Chat) channel or the ML (Mailing List) or attends a sprint.