**VERSION CONTROL**

Versioncontrol is a system that records changes to a file or set of files over time so that you can recall specific versions later. Many people version control is to save the particular file in other directories but it leads to several issues like by mistake you may modify or change the file you don’t mean to or files got deleted To deal with this issue, programmers long ago developed local VCSs that had a simple database that kept all the changes to files under revision control. But it only saves our versions on our local computer. When we work in a team, we need to see others versions or may be modify them if needed. We need to collaborate with other members in our team. To resolve this issue version control is majorly categorized into two types they are

1. CentralizedVersionControlSystems.
2. Distributed Version Control Systems.

**Centralized Version Control System:**

These systems such as CVS, Subversion, and Perforce have a single server that contains all the versioned files, and a number of clients that check out files from that central place. For many years, this has been the standard for version control. Everyone on the team can know what others are doing. It also stores every version of every file. This system also has downside if there is any problem in server people can’t work and upload their versions.

**Distributed Version Control Systems:**

To resolve above issue this system step in. In a DVCS such as Git, Mercurial, Bazaar clients don’t just check out the latest snapshot of the files; rather, they fully mirror the repository, including its full history. Thus, if any server dies, and these systems were collaborating via that server, any of the client repositories can be copied back up to the server to restore it. Every clone is really a full backup of all the data.