***MODULE 2***

***Version control systems*** Version control systems are a category of software tools that help in recording changes made to files by keeping a track of modifications done in the code.

As we know that a software product is developed in collaboration by a group of developers they might be located at different locations and each one of them contributes to some specific kind of functionality/features. So to contribute to the product, they made modifications to the source code(either by adding or removing). A version control system is a kind of software that helps the developer team to efficiently communicate and manage(track) all the changes that have been made to the source code along with information like who made and what changes have been made. A separate branch is created for every contributor who made the changes and the changes aren’t merged into the source code unless all are analyzed as soon as the changes are green signaled they merged to the main source code. It not only keeps source code organized but also improves productivity by making the development process smooth.

Basically Version control system keeps track on changes made on a particular software and take a snapshot of every modification. Let’s suppose if a team of developer add some new functionalities in an application and the updated version is not working properly so as the version control system keeps track of our work so with the help of version control system we can omit the new changes and continue with the previous version.

**Benefits of the version control system:**

1. Enhances the project development speed by providing efficient collaboration,
2. Leverages the productivity, expedites product delivery, and skills of the employees through better communication and assistance,
3. Reduce possibilities of errors and conflicts meanwhile project development through traceability to every small change,
4. Employees or contributors of the project can contribute from anywhere irrespective of the different geographical locations through this **VCS,**
5. For each different contributor to the project, a different working copy is maintained and not merged to the main file unless the working copy is validated. The most popular example is **Git, Helix core, Microsoft TFS,**
6. Helps in recovery in case of any disaster or contingent situation,
7. Informs us about Who, What, When, Why changes have been made.

**Types of Version Control Systems:**

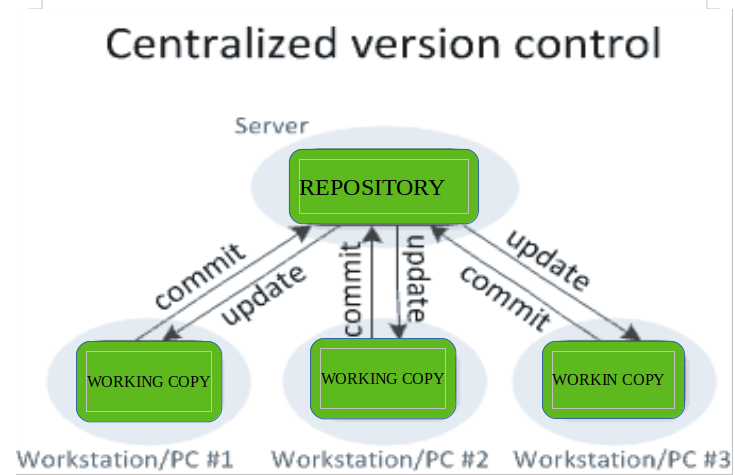
1. Local Version Control System
2. Centralized Version Control Systems
3. Distributed Version Control Systems

**Local Version Control Systems:** It is one of the simplest forms and has a database that kept all the changes to files under revision control. RCS is one of the most common VCS tools. It keeps patch sets (differences between files) in a special format on disk. By adding up all the patches it can then re-create what any file looked like at any point in time.

**Centralized Version Control Systems:** Centralized version control systems contain just one repository globally and every user need to commit for reflecting one’s changes in the repository. It is possible for others to see your changes by updating.

Two things are required to make your changes visible to others which are:

* You commit
* They update



The **benefit** of CVCS (Centralized Version Control Systems) makes collaboration amongst developers along with providing an insight to a certain extent on what everyone else is doing on the project. It allows administrators to fine-grained control over who can do what.

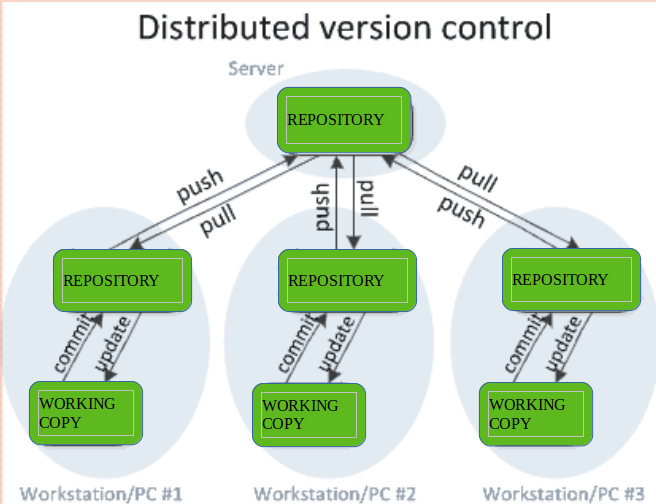
It has some **downsides** as well which led to the development of DVS. The most obvious is the single point of failure that the centralized repository represents if it goes down during that period collaboration and saving versioned changes is not possible. What if the hard disk of the central database becomes corrupted, and proper backups haven’t been kept? You lose absolutely everything.

**Distributed Version Control Systems:** Distributed version control systems contain multiple repositories. Each user has their own repository and working copy. Just committing your changes will not give others access to your changes. This is because commit will reflect those changes in your local repository and you need to push them in order to make them visible on the central repository. Similarly, When you update, you do not get others’ changes unless you have first pulled those changes into your repository.

To make your changes visible to others, 4 things are required:

* You commit
* You push
* They pull
* They update

The most popular distributed version control systems are Git, and Mercurial. They help us overcome the problem of single point of failure.



**Purpose of Version Control:**

* Multiple people can work simultaneously on a single project. Everyone works on and edits their own copy of the files and it is up to them when they wish to share the changes made by them with the rest of the team.
* It also enables one person to use multiple computers to work on a project, so it is valuable even if you are working by yourself.
* It integrates the work that is done simultaneously by different members of the team. In some rare cases, when conflicting edits are made by two people to the same line of a file, then human assistance is requested by the version control system in deciding what should be done.
* Version control provides access to the historical versions of a project. This is insurance against computer crashes or data loss. If any mistake is made, you can easily roll back to a previous version. It is also possible to undo specific edits that too without losing the work done in the meanwhile. It can be easily known when, why, and by whom any part of a file was edited.