EXPERIMENT: 8

(2K17/SE/79 PARV GUPTA)

AIM: Why is version control important? How many types of version control systems are there? Demonstrate how version control is used in a proper sequence (stepwise).

First what is version control?

In software engineering, version control (also known as revision control, source control, or source code management) is a class of systems responsible for managing changes to computer programs, documents, large web sites, or other collections of information. Version control is a component of software configuration management.

This includes version control software, version control systems, or version control tools. Version control is a component of software configuration management. It's sometimes referred to as VCS programming.

As mentioned above, version control is sometimes referred to as revision control or source control. It’s an important component of software configuration management.

Why version control is important?

Version control is important to keep track of changes — and keep every team member working off the latest version. You should use version control software for all code, files, and assets that multiple team members will collaborate on.

It needs to do more than just manage and track files. It should help you develop and ship products faster. This is especially important for teams practicing DevOps.

That’s because using the right one:

- Improves visibility.
- Helps teams collaborate around the world.
- Accelerates product delivery.

When you work on a development team, you may be touching similar parts of the code throughout a project. As a result, changes made in one part of the source can be incompatible with those made by another developer working at the same time.

Version control helps solve these kinds of problems and provides:

- A complete history of every file, which enables you to go back to previous versions to analyze the source of bugs and fix problems in older versions.
- The ability to work on independent streams of changes, which allow you to merge that work back together and verify that your changes conflict.
- The ability to trace each change with a message describing the purpose and intent of the change and connect it to project management and bug tracking software.

How many types of version control systems are there?

There are three types of version control: centralized and distributed.

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:
With centralized version control systems, you have a single “central” copy of your project on a server and commit your changes to this central copy. You pull the files that you need, but you never have a full copy of your project locally. Some of the most common version control systems are centralized, including Subversion (SVN) and Perforce.

Distributed version control:
With distributed version control systems (DVCS), you don't rely on a central server to store all the versions of a project’s files. Instead, you clone a copy of a repository locally so that you have the full history of the project. Two common distributed version control systems are Git and Mercurial.

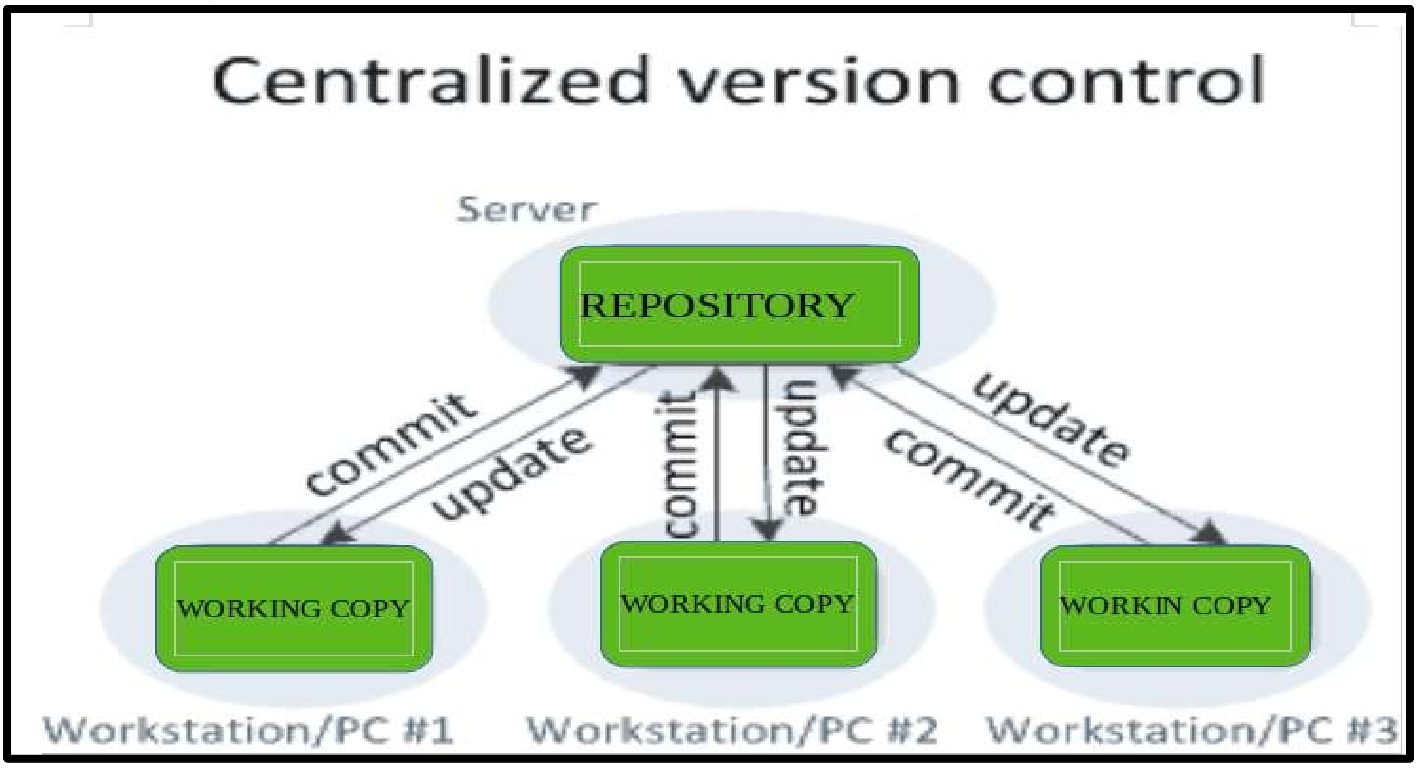
While you don't have to have a central repository for your files, you may want one "central" place to keep your code so that you can share and collaborate on your project with others. That's where Bitbucket comes in. Keep a copy of your code in a repository on Bitbucket so that you and your teammates can use Git or Mercurial locally and to push and pull code.

Here are a few of the most popular types of VCS:
- Helix Core (Perforce)
- Git
- SVN (Subversion)
- ClearCase

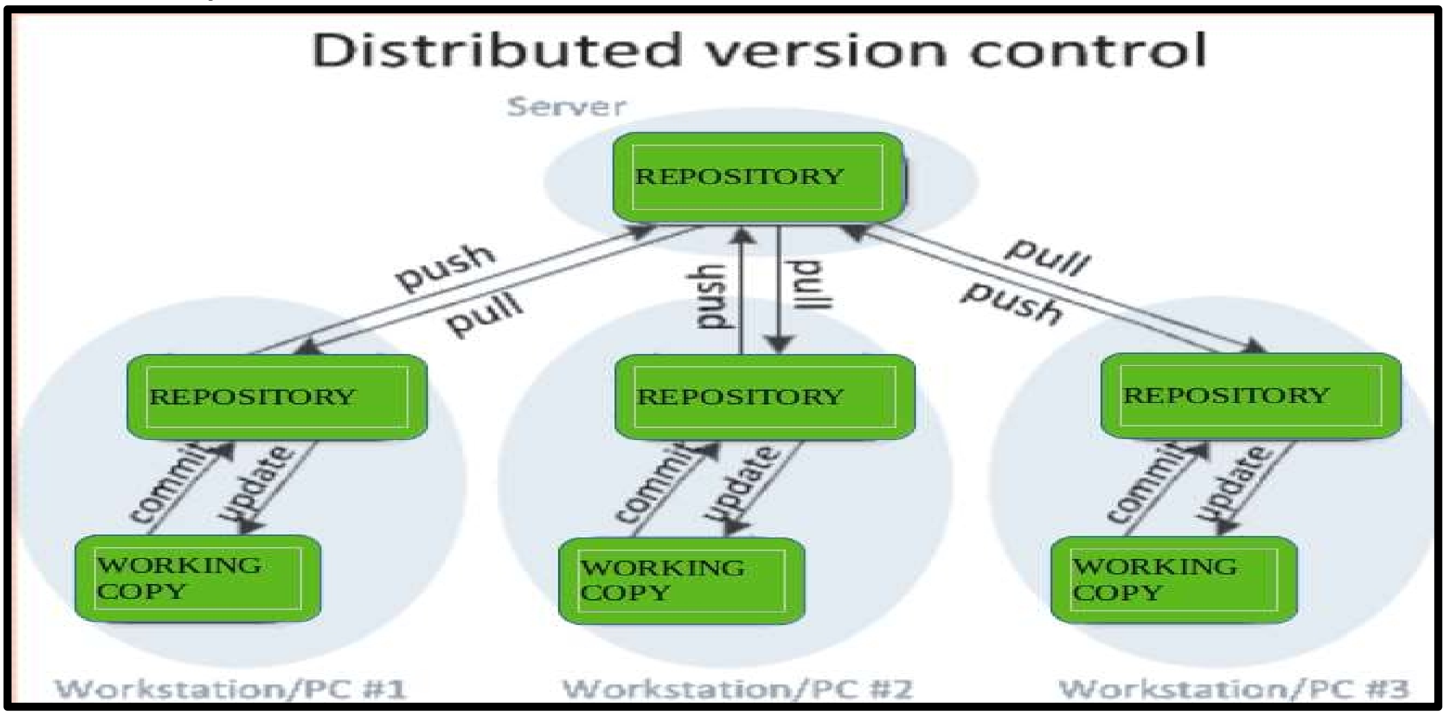
- Mercurial
- TFS (Team Foundation Server

Version control System stepwise proper sequence:

1. Proper stepwise sequence to use VCS in Centralized Version Control System.



2. Proper stepwise sequence to use VCS in Distributed Version Control System.


Distributed version control
Server
REPOSITORY
push
pull
push
pull
pull
push
REPOSITORY
REPOSITORY
REPOSITORY
commit
update
commit
update
commit
update
WORKING COPY
WORKING COPY
WORKING COPY
Workstation/PC #1
Workstation/PC #2
Workstation/PC #3