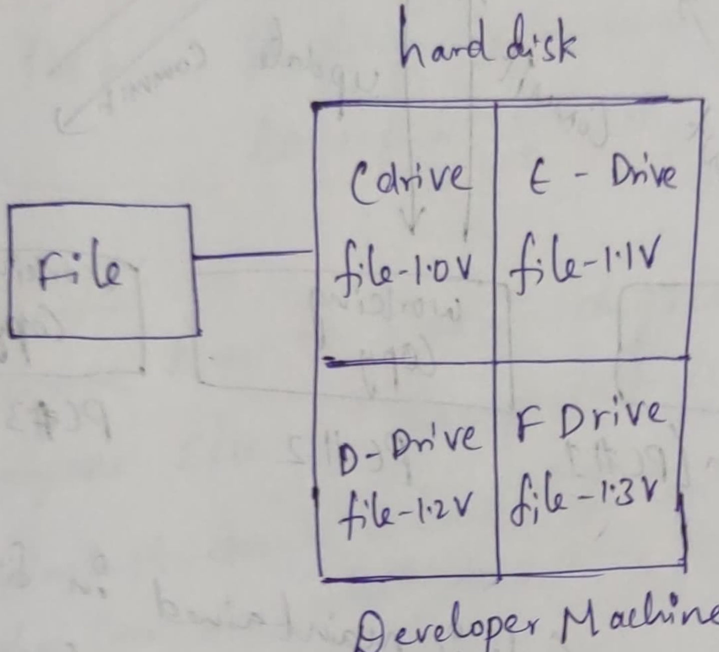
21/10/22

@ local Version Control System :-

→ it is used to maintain the file version and retreive the files based on the Specific Version.



Drawbacks :-

① it is Easy to forget in which driver you are in and accidentally write data to the wrong file. (or) Copy from other files.

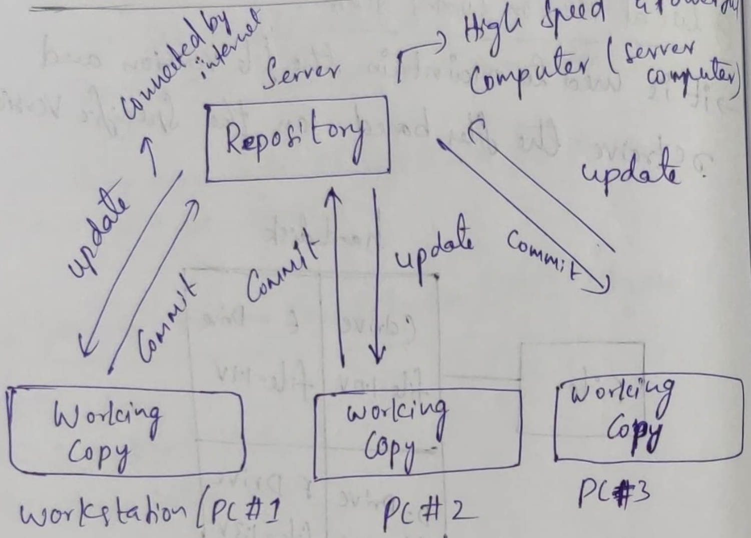
② loss of data due to Virus (or) problem in Hard disk (or) Local Machine.

③ there is possible loss of secured data.

④ By Mistake we Can delet few files also

→ To over come the drawbacks of local (VCS) we have u Centralized VCS.

⑥ Centralized Version Control System :-



→ all developers Code is maintained in single Computer (or) Repository (on ~~Sever~~ Server Computer

→ it is Called Centralized server.

→ Centralized Version Computer :-

① Developer's Can Collaborate the Code in one repository and do the change.

Eg:- Subversion, Perforce... → Centralized Version software

② Centralized Version Sever will have Single Server that Contains all the Version files.

③ for many ~~version~~ years this has been the standard Version Control System.

④ More no of developers Would Connect to CVS to checkout the files.

Checkout → taking the Code from repository to local machine.

Push → Sending Code from local machine to repository (VCS).

Advantages

① Everyone know to Certain degree What Everyone Else on the project is doing.

② Administrator will have full Control over who Can do what and it is Easier to Manage.

→ Admin Manages the Server Computer.

Disadvantages

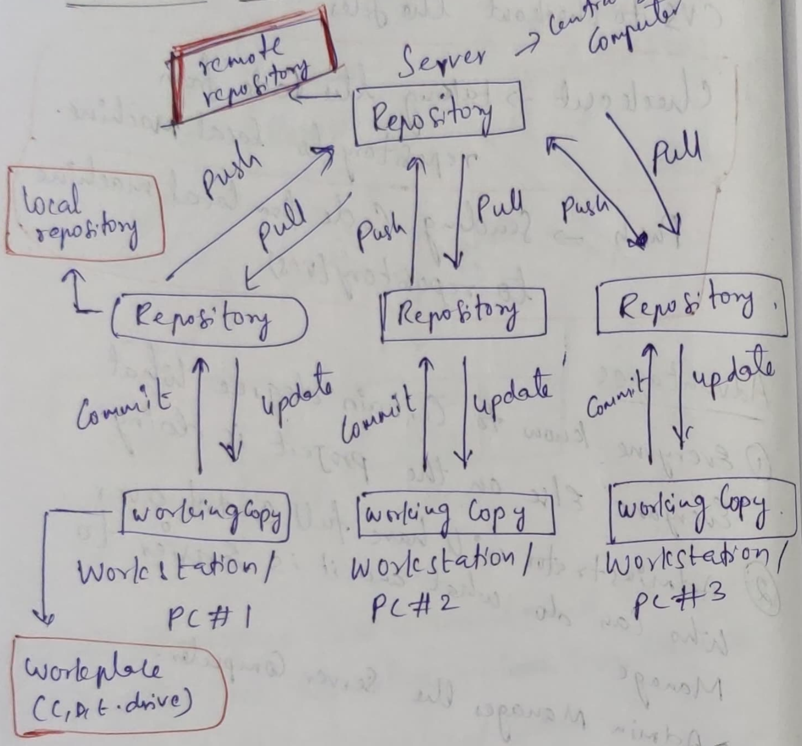
① Single point of failure (SPF) ~~would~~ would represent the Centralized system.

② if the server goes down due to network traffic, Collabration will not possible (or) save changes to server

③ if hard disk of Centralized System gets Corrupted and proper backup haven't been taken then there is every possiblity of loss of data.

④ there is chance of System Crash.

① Distrubted Version Control System :-



→ developer Maintain one Local Repository.
↳ back up Copy.

→ Even after N/w failer developer Can Continue his Work.

→ Eg : Github, bitbucket, gitlab, Mercurial, Darcs, Baesaretc.
↳ commonly used

① if the main Server goes off, still there is a Local repo which would have maintained the Copy of repo where the Entire Code is available (history of Versions).

② if the remote repository is down, then developers Can do changes in the Local repository and when the main repository is up the Code can be pushed to remote repository from Local repository.

Note: ① In LVCS and in CVCS getting up the complete history of changes is not Possible.

Version
file → 1.0V
file → 1.1V
file → 1.2V
file → 1.3V

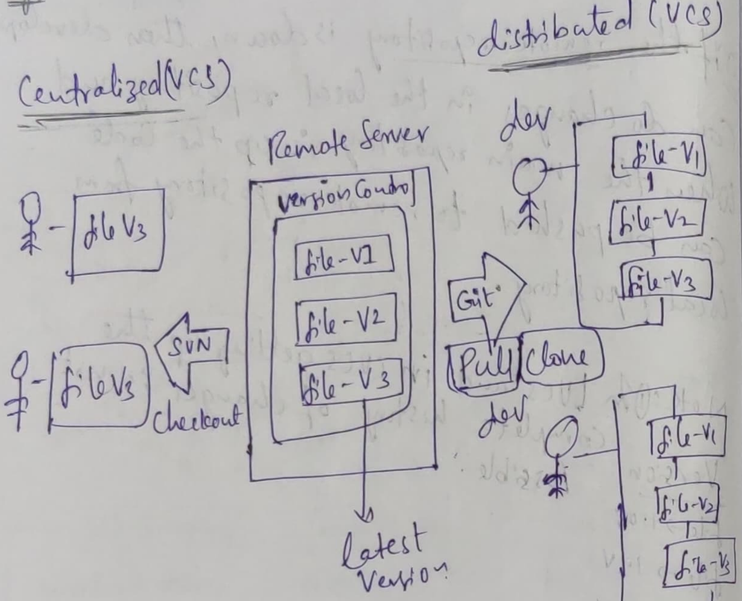
② it is possible to get only the latest Version, but not the Entire history.

Eg: SVN [Push will ~~happ~~ not happen w.r.t Version rather push will happen only with the latest ~~version~~ change]

③ In distribute (VCS) developers will not ~~get~~ only get the latest Version but also the Compilete history.

④ Push will not only happens with latest Snapshot of the files rather they will push the old files also.

Eg:-



③ Installation of git Software :-

→ ① git download.

there are 2 type of git Software

① Git Server

② Git client.

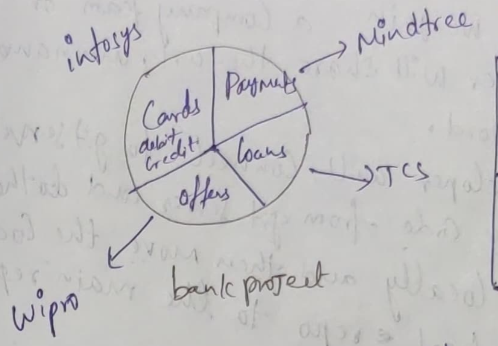
① Git Server:-→ it is a repository.

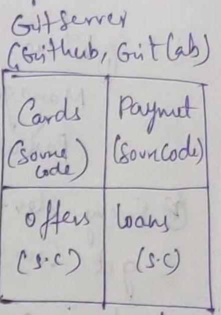
→ it is the largest host of Source Code in the world.

→ it is used to store/Manage the Source Code of project.

→ Some of git Server tools are → gitlab. → Github. → Bitbucket

Eg:-





→ All Code is kept in one repository.

→ Git Server (Github, Gilab)

→ git url will be Same for all the developers but username and password will different for all developers.

Eg ② How to Connect to github? → Sample

url → http://repo.bank:9999

~~co~~ url :→

username→

Password→

Dev1:- url → http:///repo.bank!9999/Projects/cards

username: Ashish

PW : **XXX

Dev2:- url→ Same

username: Naveen

P.W : XXXXX

Note :- ① When we join a Company team or Manager will share the url, username, Password.

② Every developer will Connect to gitserver and get Source Code from git server and do the Changes locally and then move the Code from the local repo to the main repo with the version.

③ Git server physical location where it is Installed Can't be Seen it would be installed on the Cloud platform like Aws, Azure (or) on any dataCenters.

③ where should we provide url, username and password?

→ We need git client to provide username, url and password.

Git Client :-

→ it is a tool which is used to Connect to our gitserver

→ if we install git client (git s/w) we get following tools for free

ⓐ git bash ⇒ linux Commands are required.

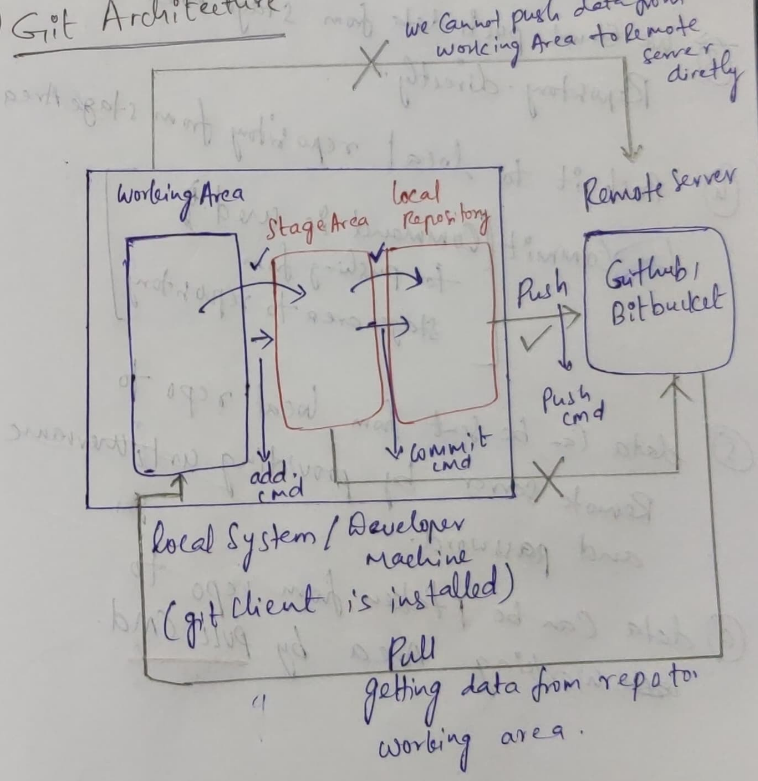
ⓑ git gui ⇒ Graphical user interface where all the actions will be done through clicks.

ⓒ git cmd :⇒ Command line tools where developers should provide url, username and password.

① git :- Client tool where client should provide url, username and password.

② github :- Server Software where repositores will be maintained.

④ Git Architecture


we Cannot push data from working Area to Remote server diretly
Working Area
Stage Area
local Repository
Remote Server
Github / Bitbucket
Push
Push cmd
add cmd
commit cmd
local System / Developer Machine
(git client is installed)
Pull
getting data from repo to working area.

→ git Architecture follows 3 regions

(i) working area:- developer will write Code and keep it in local drives. (C, E, F drive)

→ it is a place developers Maintain there Source Code.

([illegible]) ~~stage area~~

① data Cannot push directly from working Area to Remote server.

② Add work Space to stage Area.
↳ by Add cmd.

③ We Cannot push data from stage Area to Repository directly.

④ Push it to local repository from stage Area.
↳ Commit [Command is used for pushing from stage-area to repository]

⑤ data Can be sent from local repo to Remote server by providing url, username and password.

⑥ data Can be taken from repo to ~~[illegible]~~ working Area by PULL Cmd.

ⓑ stage area:- once the Code is ready, then it will be added to stage area (indication to git software)

ⓒ local repository:- once the Code is in Stage area, we Commit it to the local repository with some standard message, from local repository we "push" to main repository by providing url, username and password.