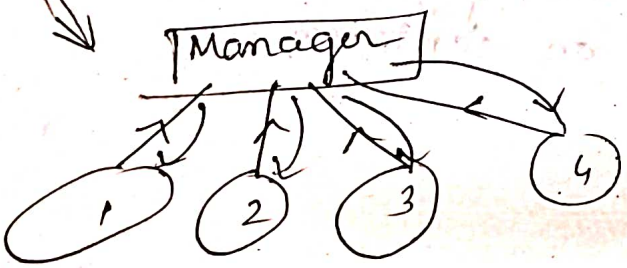
Git

Software configuration management
or
Source code management

One person write a code and sends it to next person who will merge this code with his code and sends it to next code

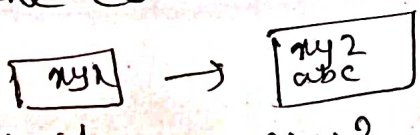
problems

- whether codes are compatible or not
- whether they can work together or not



take each code and combine it

Source code management manages the versions of all the code.

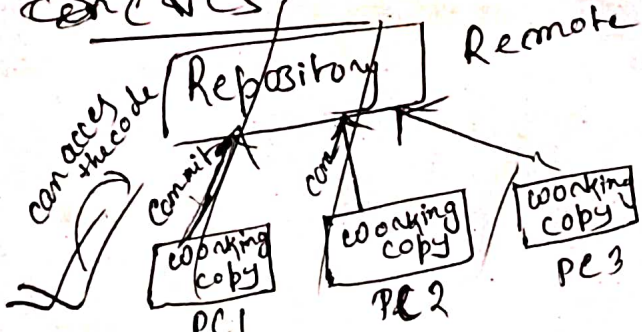


But here also two types of management system

Source Code Management

- Centralized version control System (CVCS)
- Distributed version control System

Centralized VCS



Each one can see each one's code

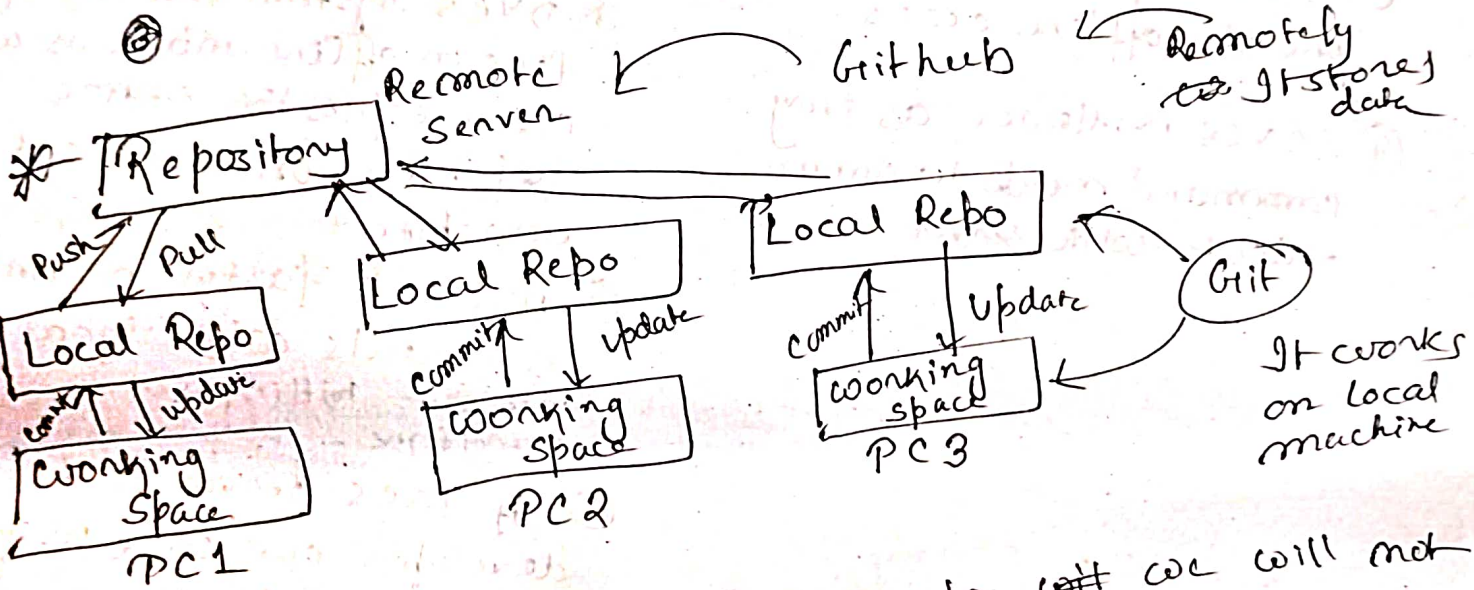
1

Drawback of CVCS

1. If remote server fails, then whole system will fail (Entire data will be lost)
2. To access remote server, you need internet
3. Slow (because of uploading, downloading)

Distributed version control system (DVCS)

1. "Git" is a distributed version control system
2. Git is a software or tool.
3.



1. If the Repository is down, then also we will not have any problem as the local repository is there
2. Version control system. For this we need git
3. If we want to share our code, we will store it in remote server

"In distributed version control system, every contributor has a local copy or 'clone' of the main Repository i.e everyone maintains a local copy of their own which contains all the files & metadata present in main repository.

4. We don't need repository

CVCS

① In CVCS, a client needs to get local copy of source from server, do the changes and commit those changes to central source on server

② Working on branches is difficult in CVCS developer often faces merge conflict

③ CVCS system do not provide offline access.

④ CVCS is slower as every command needs to communicate with server

DVCS

① In DVCS, each client can have a local repo as well and have a complete history on it. Client needs to push the changes to branch which will then be pushed to server repository.

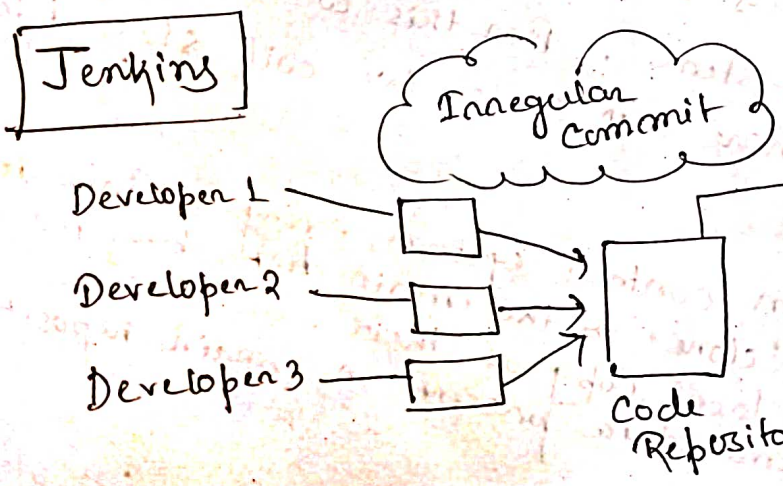
② Working on branches is easier in DVCS. Developers faces less conflict.

③ DVCS system are working fine on offline mode as a client copies the entire repository on their local machine.

④ DVCS is faster as mostly user deals with local copy without hitting server every time.

⑤ If DVCS server is down developer can work using their local copies.

Jenkins



- Developers had to wait till the entire software code was build and tested to check for errors.
- The was no iterative improvement of code and software delivery process was slow.

Jenkins is a continuous integration tool that allows continuous development, test and deployment of newly created codes.

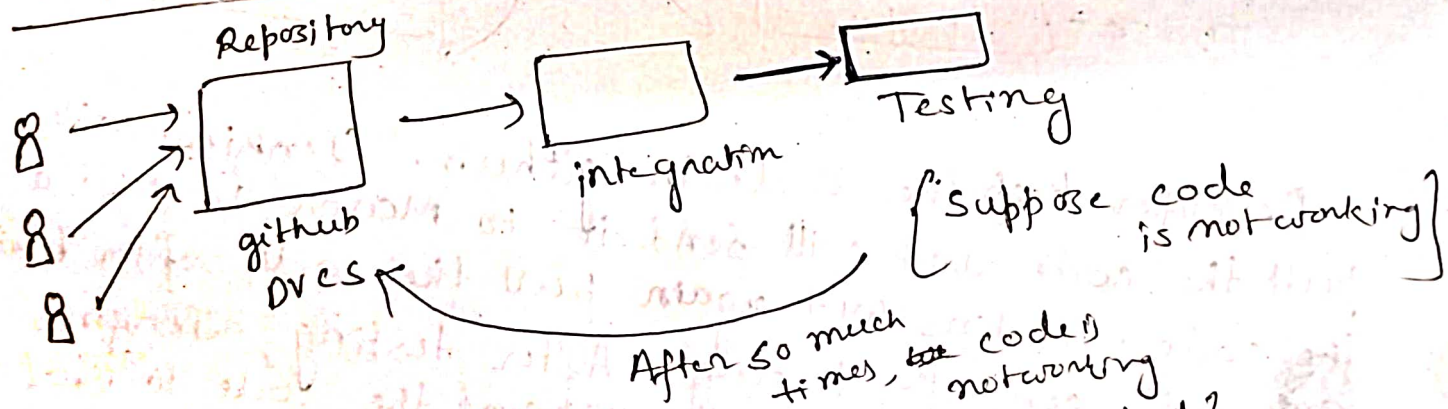
Earlier

① Developer changes to source code
② All the codes will be pulled once at a time
③ All the changes made to the code are build together.

With Jenkins

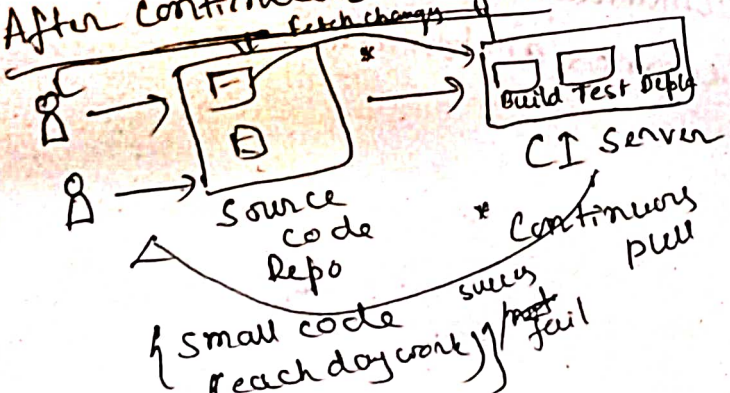
① Developer commits changes to source code
② Code is pulled whenever there is a commit made to the source code
③ All the changes made to the source code is build continuously

Before Continuous Integration



So it is lot of time consuming

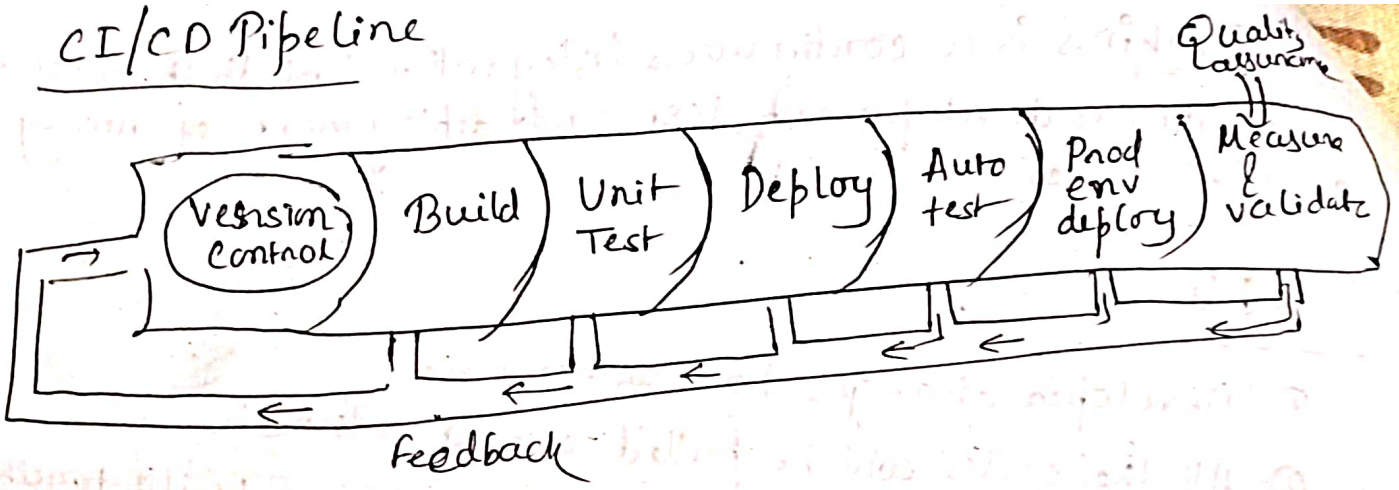
After continuous integration

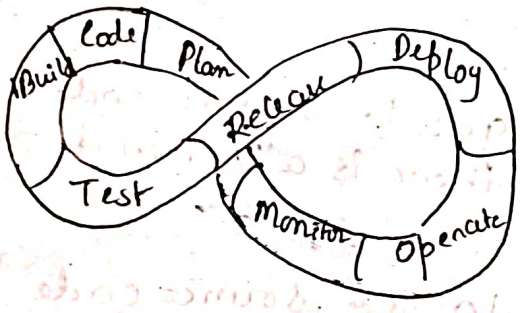


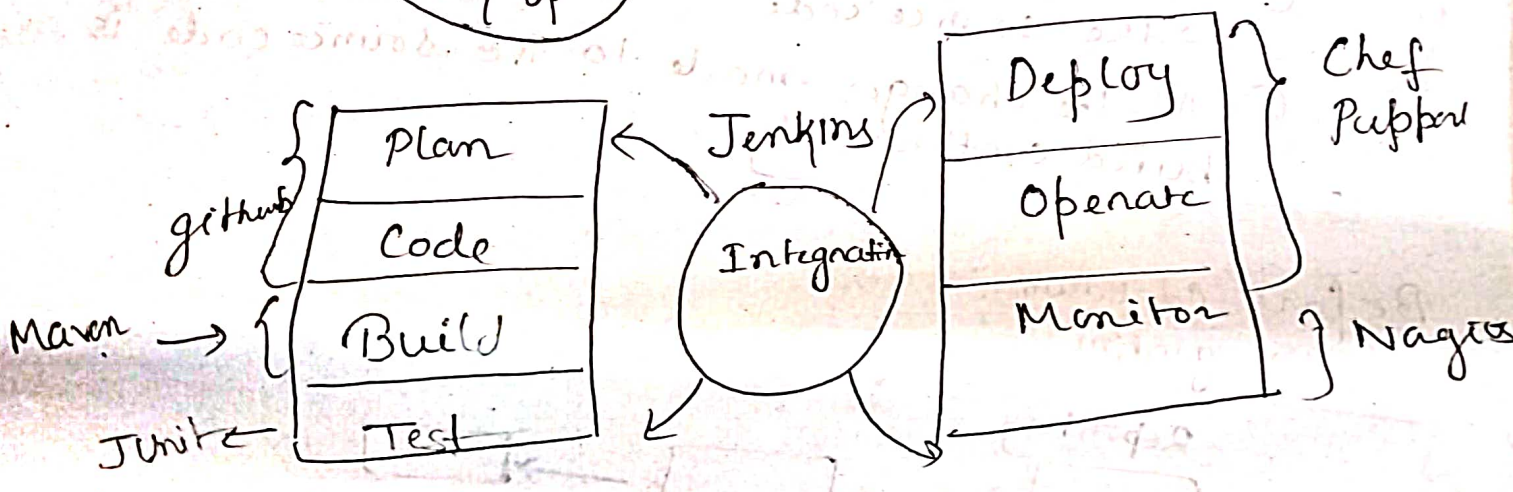
This is automated

Continuous integration = continuous Build + Continuous Testing

CI/CD Pipeline







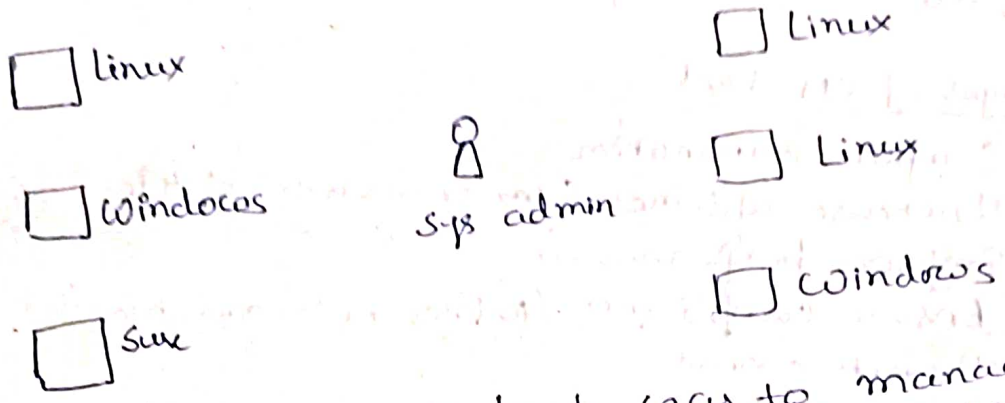
Developer keep the code in github. Jenkins will pull the code and will send it to Maven to build the code. Jenkins will again pull the code after build and will send for testing. After testing, Jenkins will again pull the code and will send the code to Chef or puppet for deployment. Then again it will send for monitoring

So Jenkins helps to communication of each stage

Because of Jenkins, it becomes automation.

Configuration Management tool - Chef

System administration is there where automation is not there.



System administrator's task was to manage different ~~task~~ server. (install, add new user etc). If small numbers of server is there then sys admin can do the job. If number of server is 500 and they need to be updated, it will be difficult for system administrator

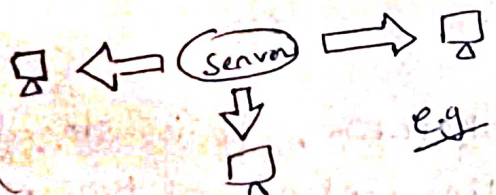
Configuration :- each and every details of your machine (server, storage)

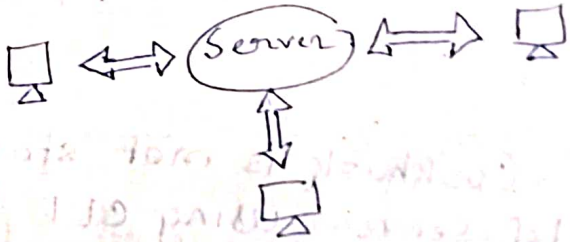
Management :- (Delete, update, create)

For this different tools.

Configuration management tool

Each and every minute detail is made.

- **Push-based**
  - Push configuration
  - Server pushes configuration to the nodes
  - Centralized
  - 

  - e.g Ansible
- **Pull based**
  - Pull conf nodes check with the server periodically and fetches the configuration from it
  - It is good when we add new machine
  - 

  - The machine will go and check server, whether same version is there or not. If not, the machine will update the software
  - e.g Chef, Puppet

Now a system's admin's role is performed by DevOps Engineer

Infrastructure as code

Using code, we create ~~code or c~~ or configure Infrastructure

{The code is present in github}

Configuration Management :- It is a method through which we automate admin tasks

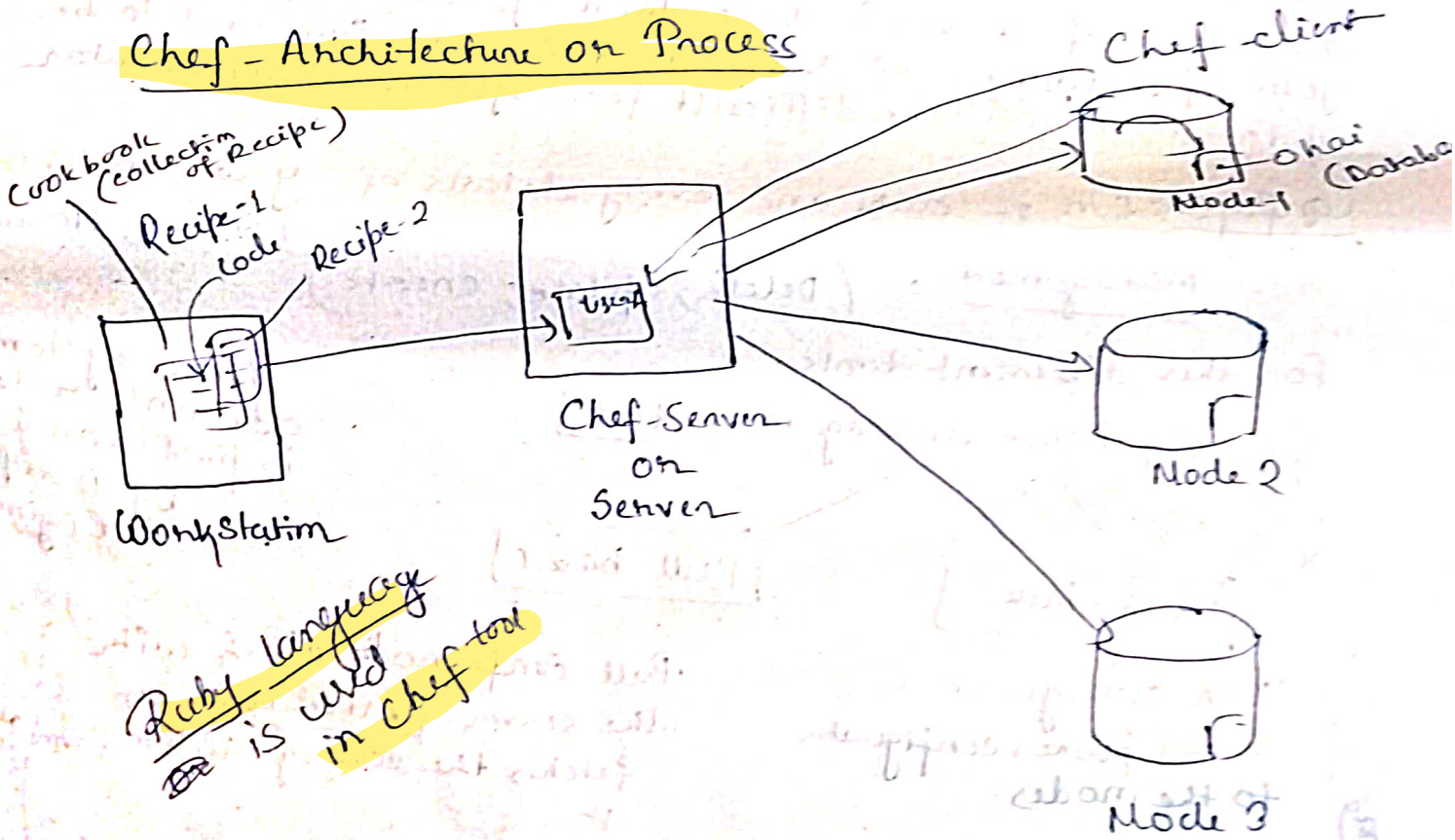
⇒ Configuration management tool turns your code into infrastructure (IAAC)

⇒ So your code would be repeatable, testable and versionable

Advantages of CM tool

→ Complete automation
→ Increase uptime { server is working }
→ Improve performance
→ Ensure compliance (follows rule and regulation)
→ Prevent errors
→ Reduce cost

Chef - Architecture or Process



Ruby language is used in Chef tool

Cookbook is not stored in workstation, it is stored in Chef-server using CLI (command line interface) tool (knife). Now it is stored in Node. The process using which server is connected to node is called **Bootstrap**. Chef client is node. Ohai is database, where node's current configuration is stored. Chef is installed in both workstation and node. The tool which is used in node is called Chef

client :- Chef client will check from Ohai and then go to server to check any software which is present is node or not. If not, it will pick it from server and install it in node. It will not pick everything, it will bring only those which is not present

Components of Chef

① **Workstation** :- (where you write code) Workstations are personal computers or virtual servers where all configuration code is created, tested or changed.

→ Devops engineer actually sits and write codes. The code is called Recipe A collection of Recipes are known as Cookbook.

→ Workstation communicate with the chef server using knife

→ Knife is a command line tool that uploads the cookbook to the server.

② **Chef-Server** :- (where you store code) The Chef-server is a middle-man between workstation and the nodes

→ All cookbooks are stored here

→ Server may be hosted locally or remote

③ **Node** (where you apply code)

Nodes are the systems that require the configuration

→ Ohai fetches the current state of the node its located in

→ Node communicate with the Chef-server using the Chef-client

→ Each node can have a different configuration required

→ chef client is installed on every node.

Knife :- Tool to establish communication among workstation, server and node. Knife is a command line tool that runs on workstation.

Chef client :- Tool runs on every chef node to pull code from chef server.

Chef client will –
- → gather current system configuration
- → Download the current system configuration from Chef-server

Ohai :- Maintain current state information of chef code.

Idempotency :- Tracking the state of system resources to ensure that the changes should not reapply repeatedly

Chef Supermarket – Where you get custom code

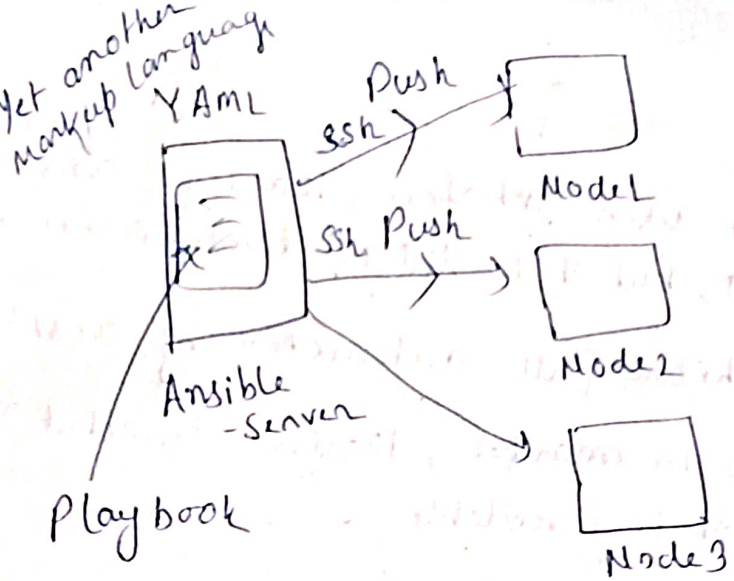
Ansible :-

- Scripting language - YAML
- configuration management tool
- Push mechanism

→ Ansible is an open source IT configuration management, Deployment and Orchestration tool. It aims to provide large productivity gains to a wide variety of automation challenges.

– Can use this tool whether your servers are in on premises or in the cloud

– It turns your code into infrastructure i.e your computing environment has some of the same attributes as your application.



- Ansible seaver will directly communicate with nodes. Here seaver pushing. That is why it is not 100% automatic. ~~Ansible~~
- Ansible is agentless (in chef we installed chef client)
  - It will communicate through SSH.
  - In chef where we write the code we call it Recipe. In Ansible where we write the code is called Playbook

Advantages

1. Ansible is free to use by everyone
2. Ansible is very consistent and lightweight and no constraints regarding the OS or underlying hardware are present.
3. It is very secure due to its agentless capabilities and open SSH security features
4. Ansible does not need any special system administrator skills to install and use it (YAML)
5. Push mechanism