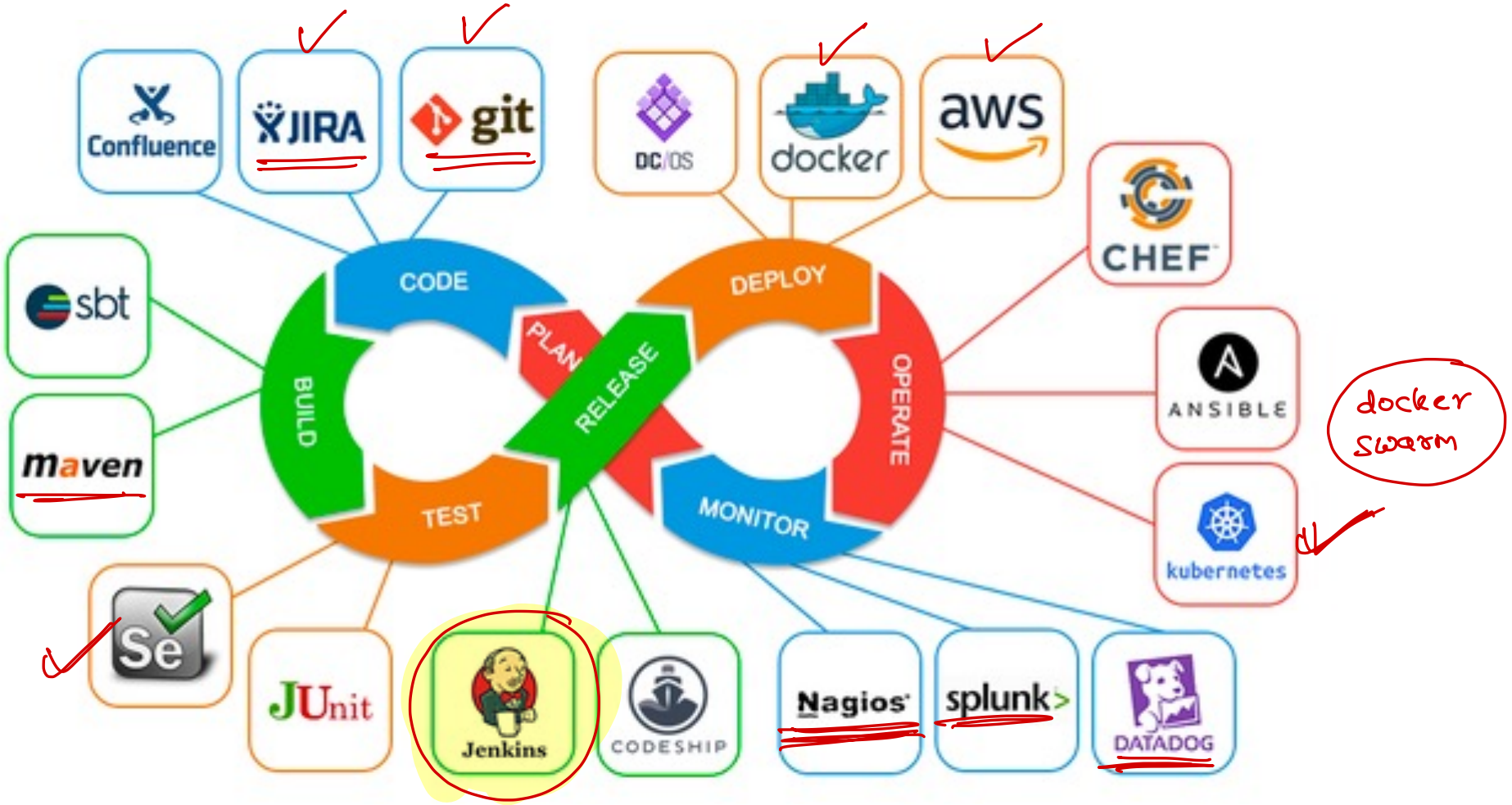
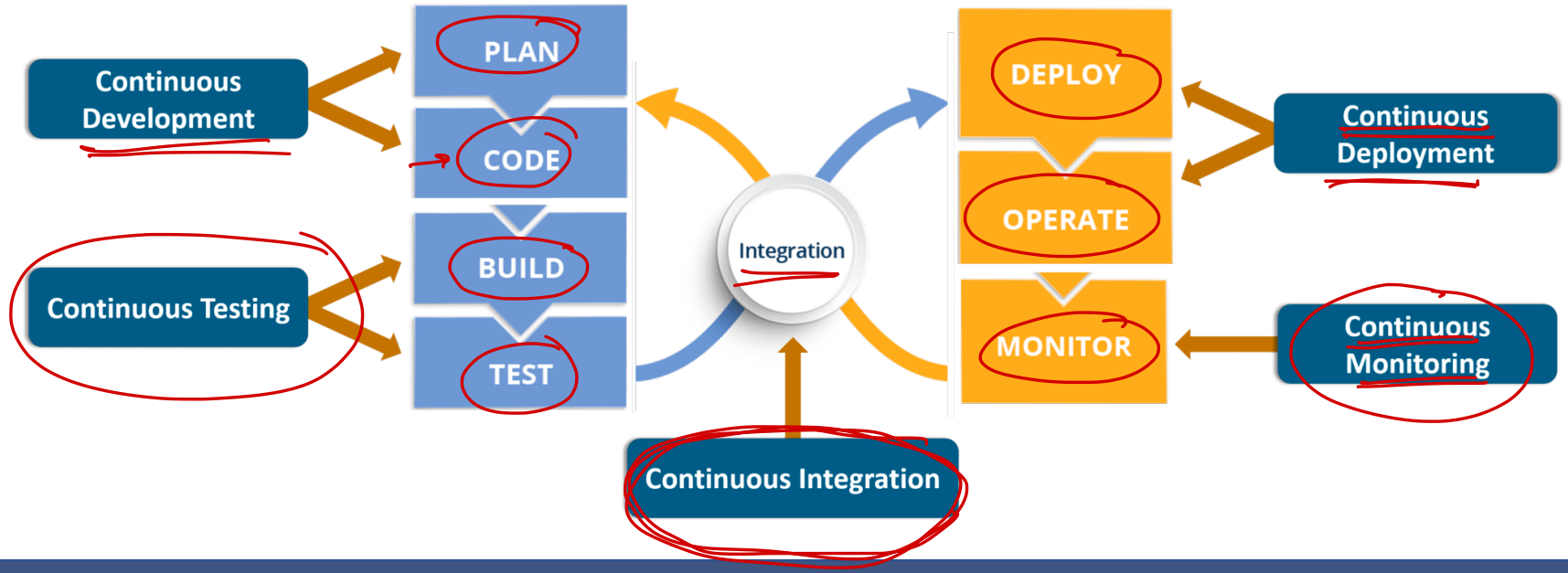
Jenkins

DevOps Lifecycle



DevOps Terminologies

- Continuous Development
- Continuous Testing
- Continuous Integration
- Continuous Delivery
- Continuous Deployment
- Continuous Monitoring

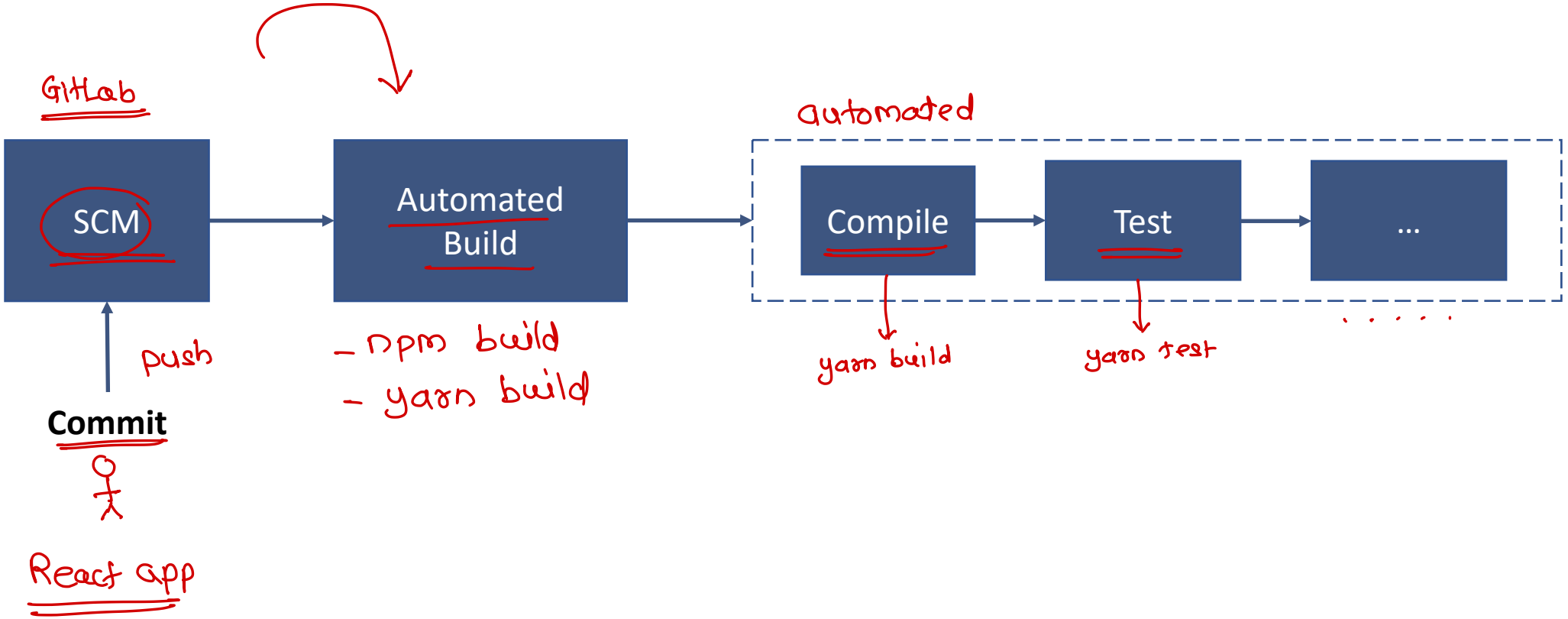


Continuous Integration

Overview

- It is the process of automating the building and testing of code, each time developer commits changes to the version control system [git]
- CI is necessary to bring out issues encountered during the integration as early as possible
- CI requires developers to have frequent builds [deployable package]
- The common practice is that whenever a code commit occurs, a build should be triggered

Continuous Integration



CI Best Practices

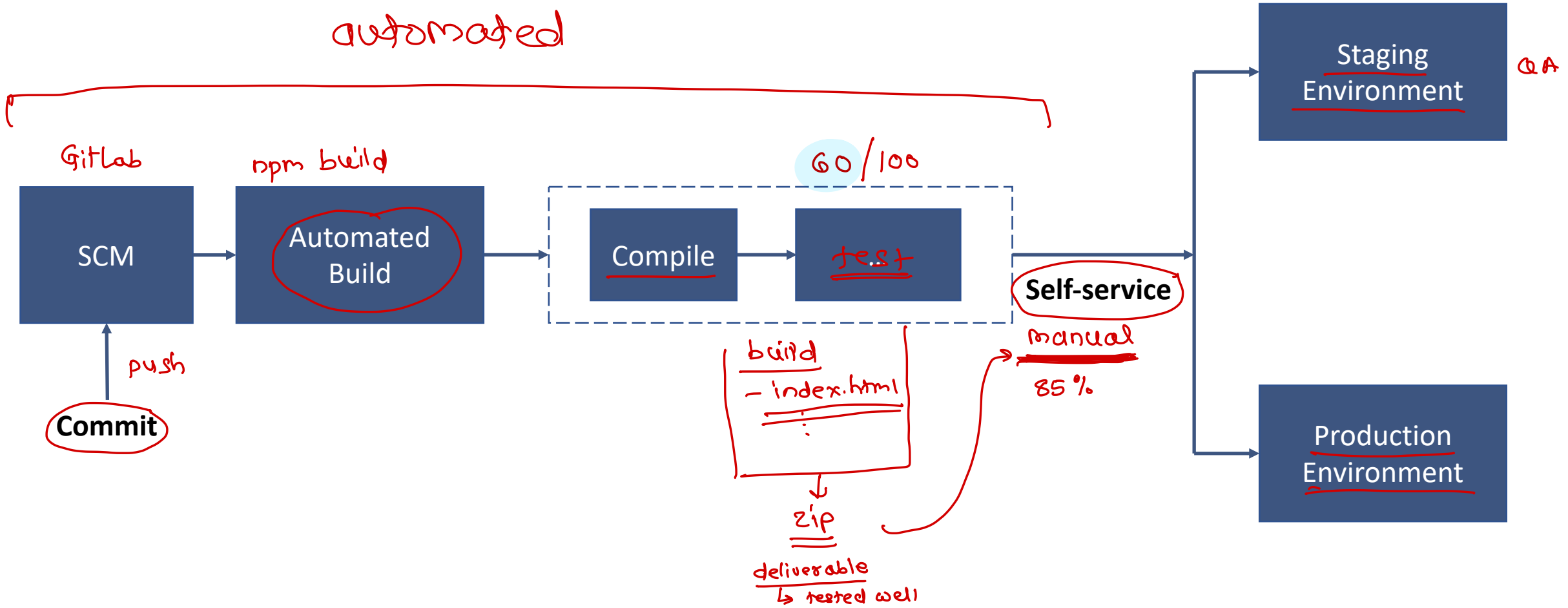
- Frequent commits [to retrieve the code, to fire builds]
- No long running branches
- Automated test execution
- Fix broken builds

master/main branch
- stable
- latest
- tested

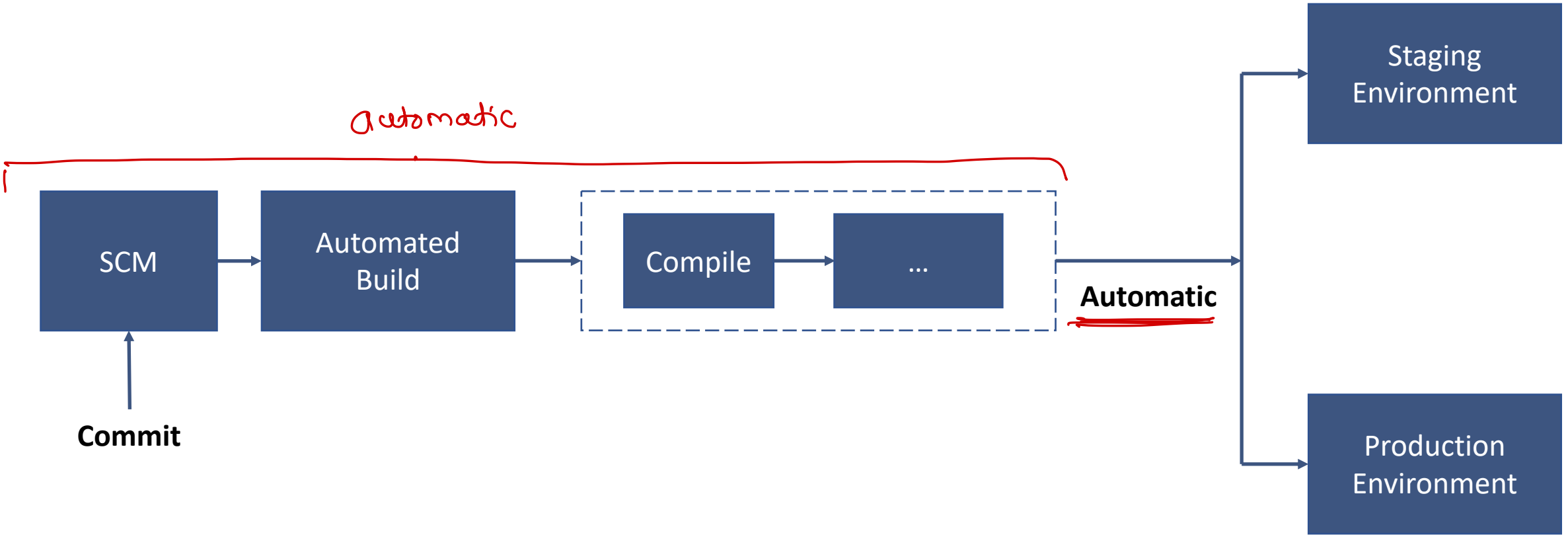
1. create branch
2. add code
3. commit the code
4. test the code
5. merge the branch into master/main

Continuous Delivery

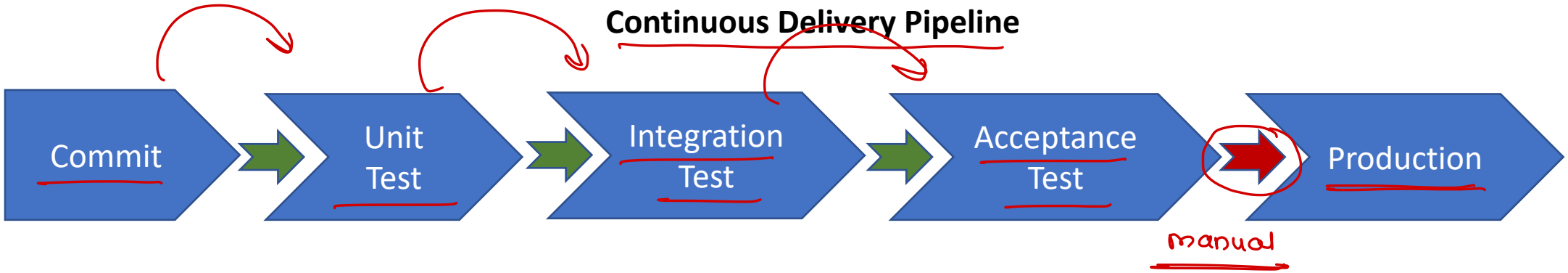
CI/CD

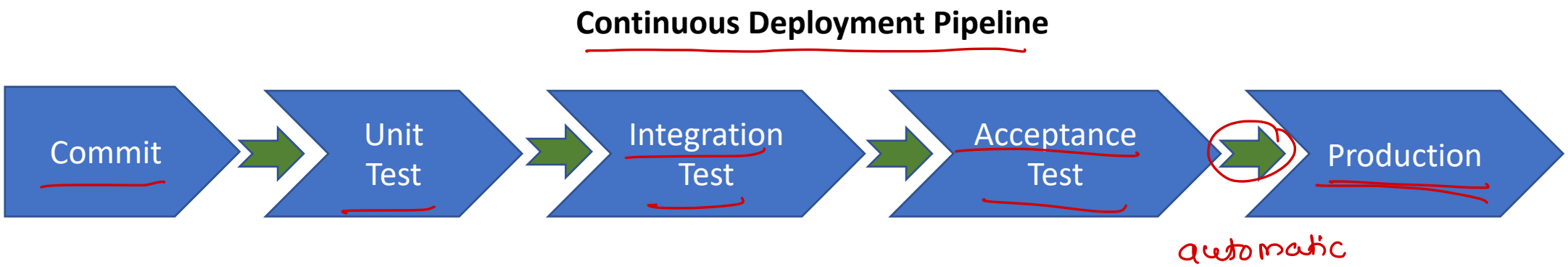


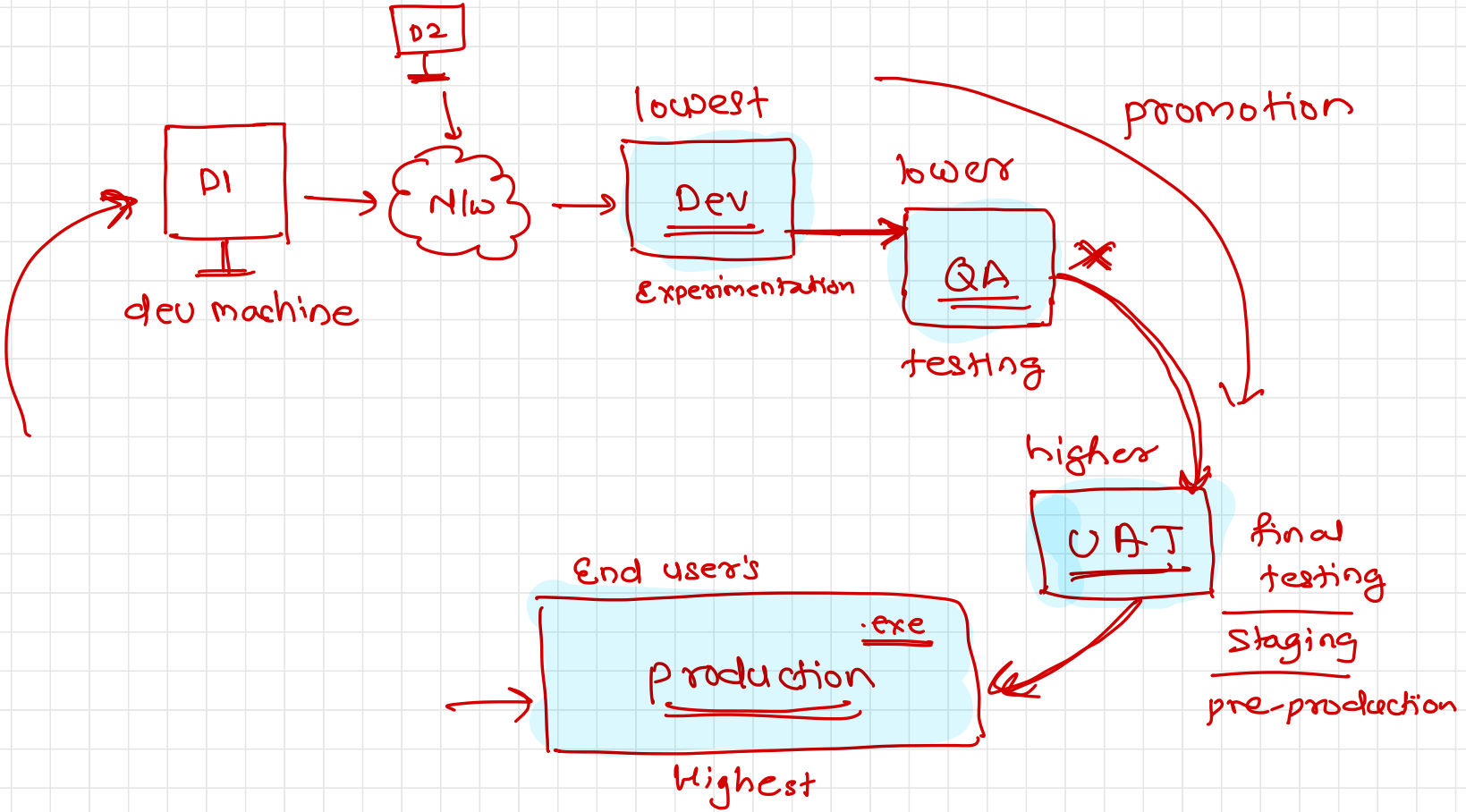
Continuous Deployment



CI/CD Pipeline






D2
D1
N/w
dev machine
lowest
Dev
Experimentation
promotion
lower
QA
testing
higher
UAT
final testing
Staging
pre-production
End user's
.exe
production
Highest

CD Best Practices

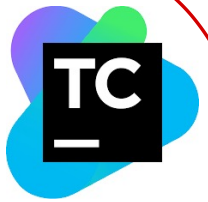
- Version control all configuration
- Stop the line immediately for a failure
- Build your binaries only once
- Deploy the same way to all the envionments

Importance

- Improves product quality
  - Improves the product quality by running the various unit test cases every time developer commits changes
- Increase productivity
  - Automating build of code saves a lot of time, thereby increasing productivity
  - Developer can utilize the time more to develop the code
- Reduces risk
  - Eliminates the potential human errors by automating test

Popular CI tools

Jenkins



TeamCity

Bamboo

GitLab CI

Travis CI

Jenkins

What is Jenkins ?

- Jenkins is a powerful application that allows continuous integration and continuous delivery of projects
- It is a free and open source application that can handle any kind of build or continuous integration

→ website
→ mobile app
→ desktop app (exe)
→ scripts

Where is it came from ?

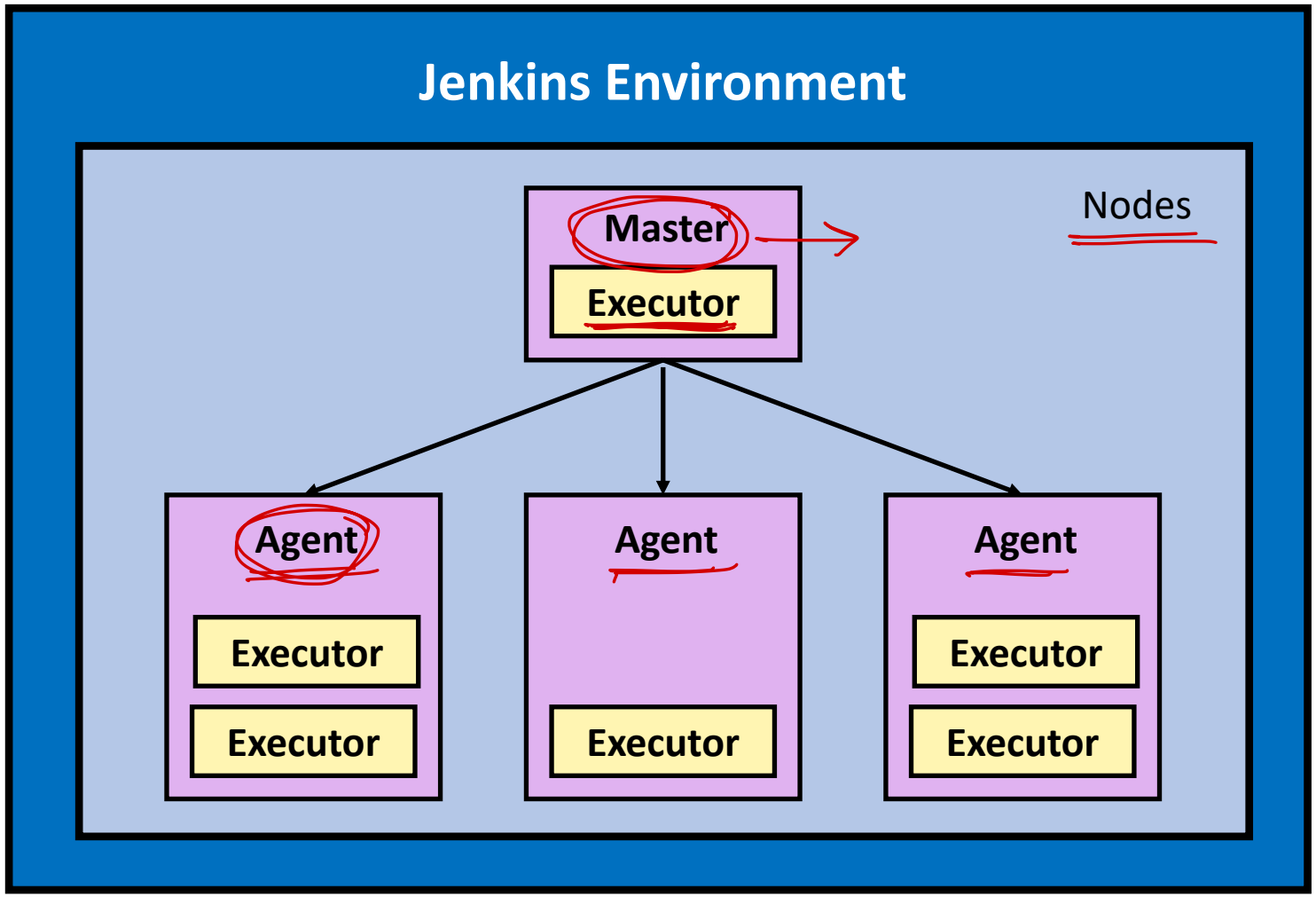
- It was first started as project Hudson at Sun Microsystems in 2004 and was first released in Feb 2005
- In 2010, Oracle acquired Sun Microsystems
- In 2011, Oracle created fork of Hudson as Jenkins, since when these two projects exist as two independent projects
- On April 20, 2016 version 2 was released with the *Pipeline* plugin enabled by default

Features

- Easy installation on different operating systems
- Supports pipelines as code that uses **domain-specific language** (**DSL**) to model application delivery pipelines as code
- Easily extensible with the use of third-party plugins
- Easy to configure the setup environment in the user interface
- Master slave architecture supports distributed builds to reduce the load on CI servers
- Build scheduling based on cron expressions * * * * *
- Shell and Windows command execution that makes any command-line tool integration in the pipeline very easy
- Notification support related to build status
  - sms
  - email
  - push
  - IM → slack

Jenkins Environment

Job
↳ project
↳ pipeline

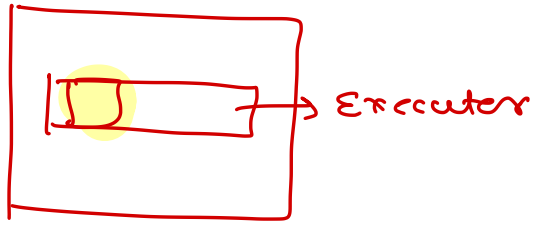


Terminologies

- **Node**
  - Node is the generic term that is used in Jenkins to mean any system that can run Jenkins jobs
  - This covers both masters and agents, and is sometimes used in place of those terms
  - Furthermore, a node might be a container, such as one for Docker
- **Master**
  - A Jenkins *master* is the primary controlling system for a Jenkins instance
  - It has complete access to all Jenkins configuration and options and the full list of jobs
  - It is the default location for executing jobs if another system is not specified
  - Master node must be present in Jenkins installation
- **Agent**
  - Is also known as Jenkins slave
  - This refers to any non-master system
  - The idea is that these systems are managed by the master system and allocated as needed, or as specified, to handle processing the individual jobs

Terminologies

- **Executor**
  - It is a slot in which to run a job on a node/agent
  - A node can have zero or more executors
  - The number of executors defines how many concurrent jobs can be run on that node
  - When the master funnels jobs to a particular node, there must be an available executor slot in order for the job to be processed immediately. Otherwise, it will wait until an executor becomes available.



Jenkins Pipeline

What is Jenkins pipeline ?

- Jenkins is, fundamentally, an automation engine which supports a number of automation patterns
- Pipeline adds a powerful set of automation tools onto Jenkins, supporting use cases that span from simple continuous integration to comprehensive CD pipelines
- It is a suite of plugins which supports implementing and integrating continuous delivery pipelines
- A continuous delivery (CD) pipeline is an automated expression of your process for getting software from version control right through to your users and customers
- Every change to your software (committed in source control) goes through a complex process on its way to being released
- This process involves building the software in a reliable and repeatable manner, as well as progressing the built software (called a "build") through multiple stages of testing and deployment

Pipeline Features

- **Code**
  - Pipelines are implemented in code and typically checked into source control, giving teams the ability to edit, review, and iterate upon their delivery pipeline.
- **Durable**
  - Pipelines can survive both planned and unplanned restarts of the Jenkins master.
- **Pausable**
  - Pipelines can optionally stop and wait for human input or approval before continuing the Pipeline run.
- **Versatile**
  - Pipelines support complex real-world CD requirements, including the ability to fork/join, loop, and perform work in parallel.
- **Extensible**
  - The Pipeline plugin supports custom extensions to its DSL [1] and multiple options for integration with other plugins.

Pipeline concepts

- **Pipeline**
  - A Pipeline is a user-defined model of a CD pipeline
  - A Pipeline's code defines your entire build process, which typically includes stages for building an application, testing it and then delivering it
  - **pipeline** block is used to create a pipeline
- **Node**
  - A node is a machine which is part of the Jenkins environment and is capable of executing a Pipeline
  - **node** block is used to define a node which can be used while executing job
- **Stage**
  - A **stage** block defines a conceptually distinct subset of tasks performed through the entire Pipeline (e.g. Build, Test, Deploy stages), which is used by many plugins to visualize or present Jenkins Pipeline status
- **Step**
  - A step in the process
  - A single task, fundamentally, a step tells Jenkins what to do at a particular point in time
  - **step** block can be used to create a step

Job

- Also known as Project or Item or Task
- It represents the steps used to build the code
- To create a new job, use option “new item”
- Project in Jenkins has different types
  - Freestyle Project
  - Pipeline
  - Multi-configuration Project
  - Folder
  - GitHub Organization
  - Multibranch Pipeline
- Demo
  - Create a freestyle project to print “hello world”

Build

- Execution of an automated task or multiple tasks
- Jenkins will run the job to create a build