Open Source Software Development

Continuous Integration (CI) - Continuous Delivery (CD)

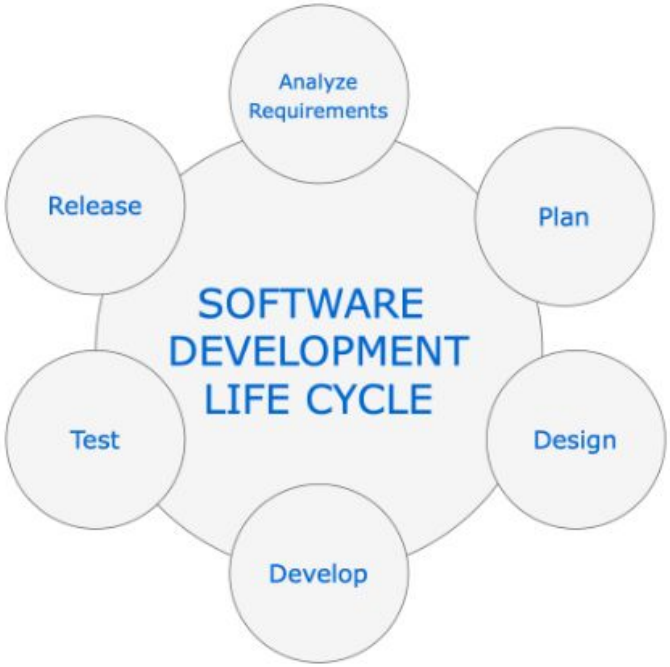
Thái Minh Tuấn - Email: minhtuan@ctu.edu.vn

Learning Objectives

- Explain the fundamental concepts of DevOps, Continuous Integration (CI), Continuous Delivery (CD), and Continuous Deployment
- Understand the basics of Continuous Delivery pipelines
- Distinguish different tools available for CI/CD and explain why Jenkins is a good choice

Software Development Life Cycle

- Software development follows a flow:
  - Identifying new features
  - Planning
  - Doing the actual development, committing the source code changes
  - Running builds and tests (unit, integration, functional, acceptance, etc.)
  - Deploying to production



Continuous Integration (CI)

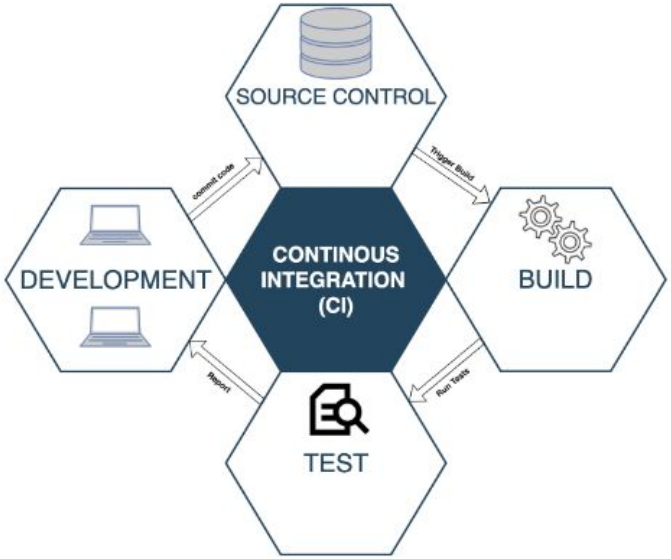
- Continuous Integration is an agile engineering practice originating from the extreme programming methodology. It primarily focuses on automated build and test for every change committed to the version control system by the developers.

  "Continuous Integration (CI) is a software development practice where members of a team integrate their work frequently; usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible".

  Martin Fowler

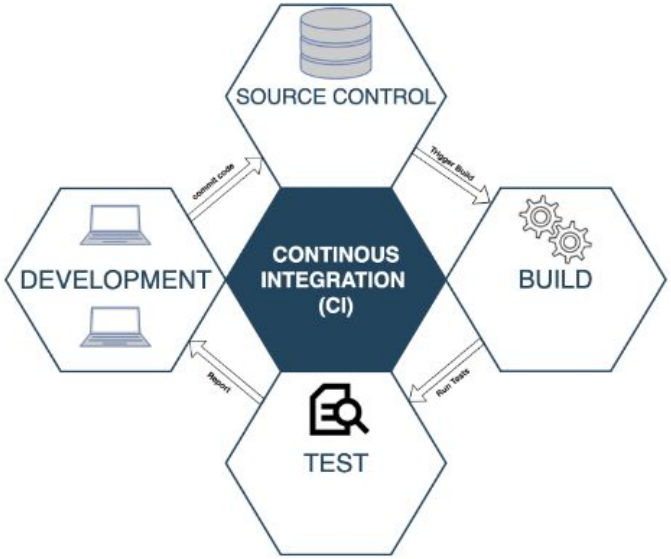
CI Implementation - Requirements (1/2)

- A version control system
  - It stores all the source code checked in by the teams, and acts as the single source of truth.
- Automated build and unit test
  - It is not sufficient if the code written by a developer works only on his/her machine.
  - Every commit that makes it to the version control system should be built and tested by an independent continuous integration server.



CI Implementation - Requirements (2/2)

- Feedback
  - Developers should get feedback on their commits.
  - Anytime a developer's change breaks the build, they can take the necessary action (example: email notifications).
- Agreement on ways of working
  - It is important that everyone on the team follows the practice of checking-in incremental changes rather than waiting till they are fully developed.
  - Their priority should be to fix any build issues that may arise with the checked-in code.



Continuous Delivery (1/2)

- A logical extension of Continuous Integration.
- Automates the full application delivery process by taking any change in code (new features, bug fixes, etc.) all the way from development (code commit) to deployment (to environments such as staging and production).
- Ensures that you are able to release new changes to your customers quickly in a reliable and repeatable manner.

Continuous Delivery is a software development discipline where you build software in such a way that the software can be released to production at any time.

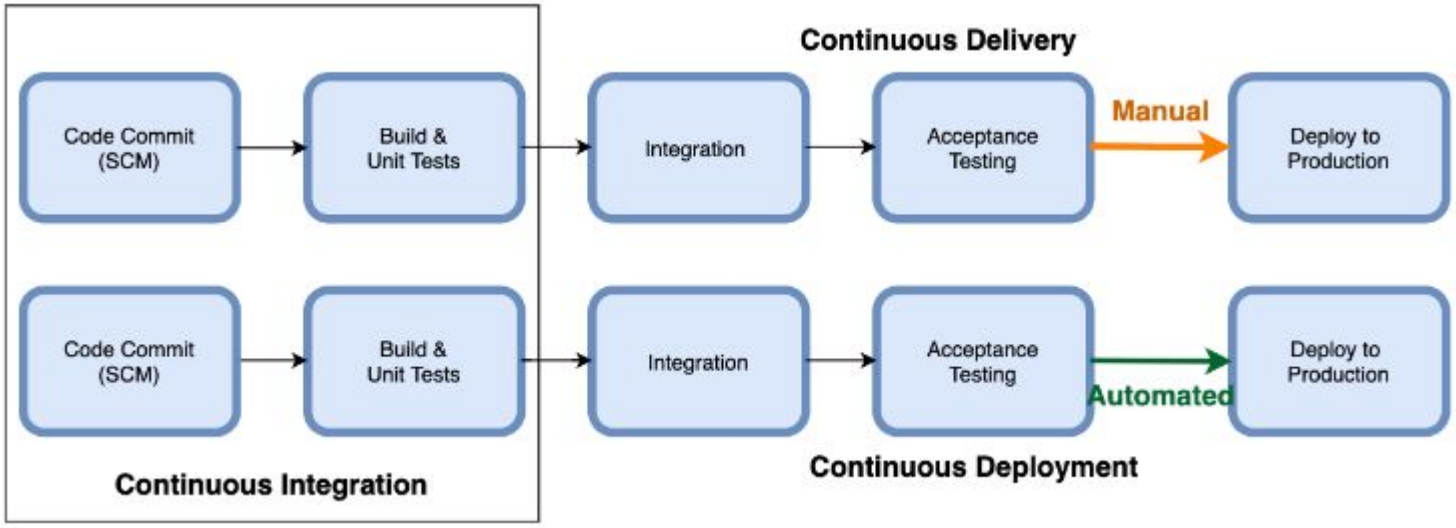
Continuous Delivery (2/2)

- To achieve continuous delivery you need:
  - A close, collaborative working relationship between everyone involved in delivery (often referred to as a DevOps culture)
  - Extensive automation of all possible parts of the delivery process, usually using a deployment pipeline
- Continuous Delivery practices will make:
  - Your overall release process painless
  - Reduce the time to market for new features, and increase the overall quality of software thereby leading to greater customer satisfaction
  - It can also significantly reduce your software development costs as your teams will prioritize releasing new features over debugging defects

Continuous Deployment

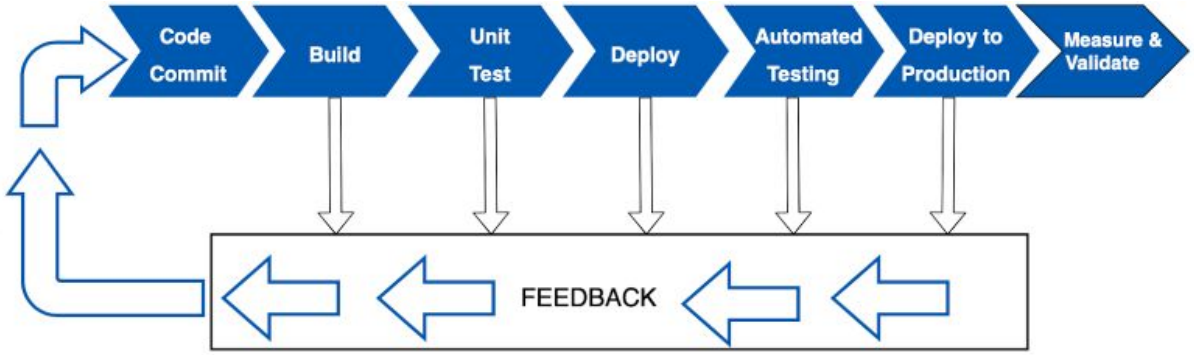
- While Continuous Delivery gives you the capability to deploy to production frequently, it does not necessarily mean that you are automating the deployment. You may need manual approval prior to deploying to production, or your business may not want to deploy frequently.
- Continuous Deployment, however, is an automated way of deploying your releases to production. You need to be doing continuous delivery in order to be able to perform automated deployment. Companies like Netflix, Amazon, Google, and Facebook automatically deploy to production multiple times a day.

Continuous Deployment



Deployment Pipelines

- Automate all the stages of your software delivery process:
  - A developer committing source code change into a version control repository.
  - The CI server detects the new commit, compiles the code, and runs unit tests.
  - The next stage is deploying the artifacts (files generated from a build) to staging or a feature test environment where you run additional functional, regression, and acceptance tests.
  - Once all the tests are successful, you are ready to deploy into production.
  - In case of failure during any stage, the workflow stops and an immediate feedback is sent back to the developer.



Benefits of CI/CD

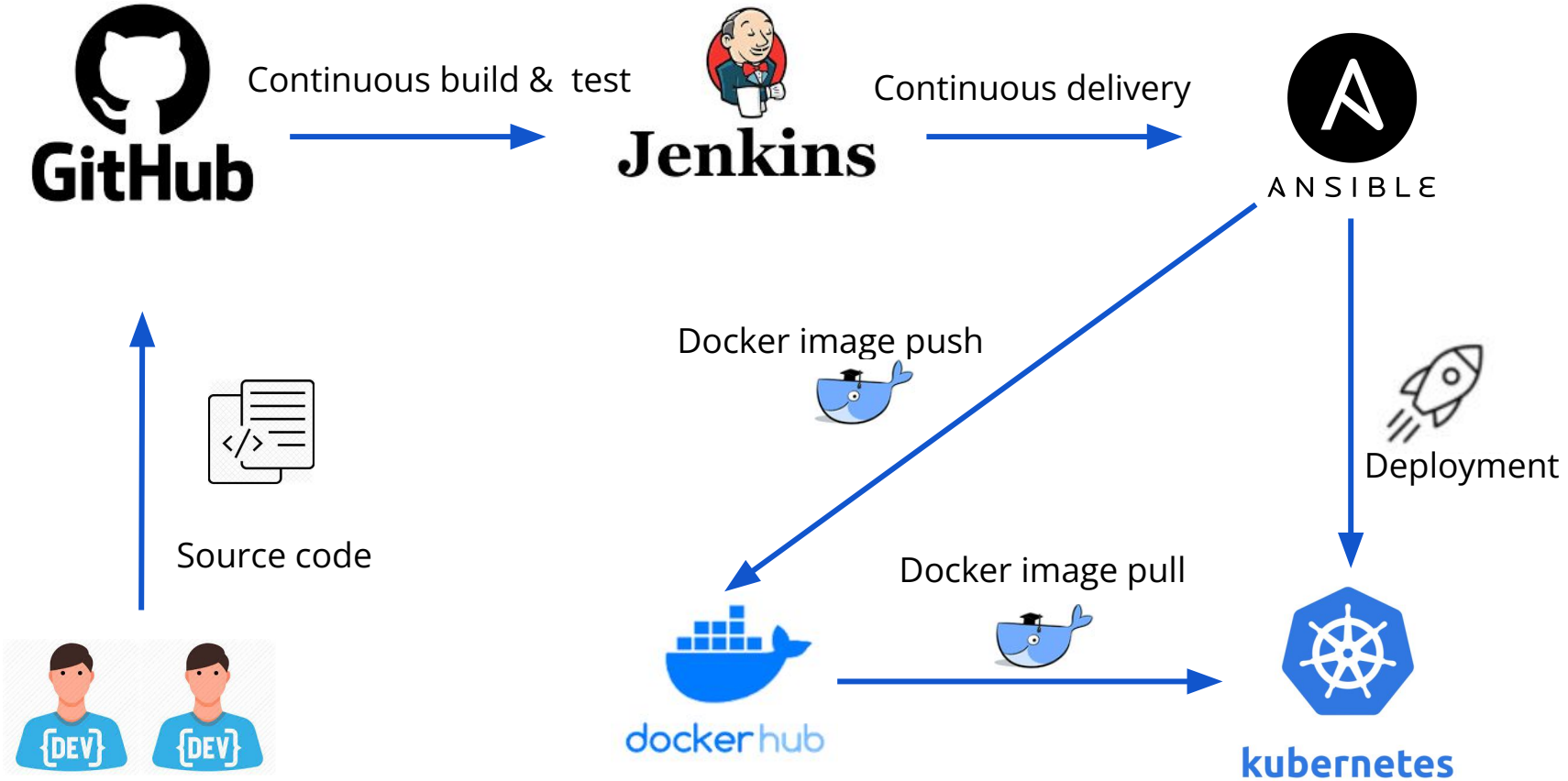
- Faster time to market
- Reduced risk
- Shorter review time
- Better code quality
- Smoother path to production
- Faster bug fixes
- Efficient infrastructure
- Measurable progress
- Tighter feedback loops
- Collaboration and communication
- Maximized creativity

https://www.jetbrains.com/teamcity/ci-cd-guide/benefits-of-ci-cd/

Tools for Deployment Pipeline

- To automate the various stages of your deployment pipeline, you will need multiple tools:
  - A version control system such as Git to store your source code
  - A Continuous Integration (CI) tool such as Jenkins to run automated builds
  - Test frameworks such as xUnit, Selenium, etc., to run various test suites
  - A binary repository such as DockerHub or Artifactory to store build artifacts
  - Configuration management tools such as Ansible
  - A single dashboard to make the progress visible to everyone
  - Frequent feedback in the form of emails, or Slack notifications.

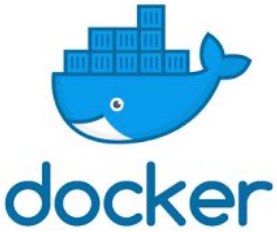
CI/CD pipeline example



Applying CI/CD pipeline for a Python Flask project

- Flask : A micro web framework written in Python

- Automatically Dockerize a Flask Project on GitHub Push with Jenkins
  - Docker: A set of platform use OS-level virtualization to deliver software in packages called containers
  - GitHub: Internet hosting for version control using Git
  - Jenkins: A free and open source automation server



Applying CI/CD pipeline for a Python Flask project