# DevOps
# lecture 6
# Continuous Integration and Continuous Delivery (CI/CD) Using Jenkins

Dr. Divya Srivastava
Assistant Professor
Bennett University, Greater Noida

February 8, 2022

# DevOps Lifecycle

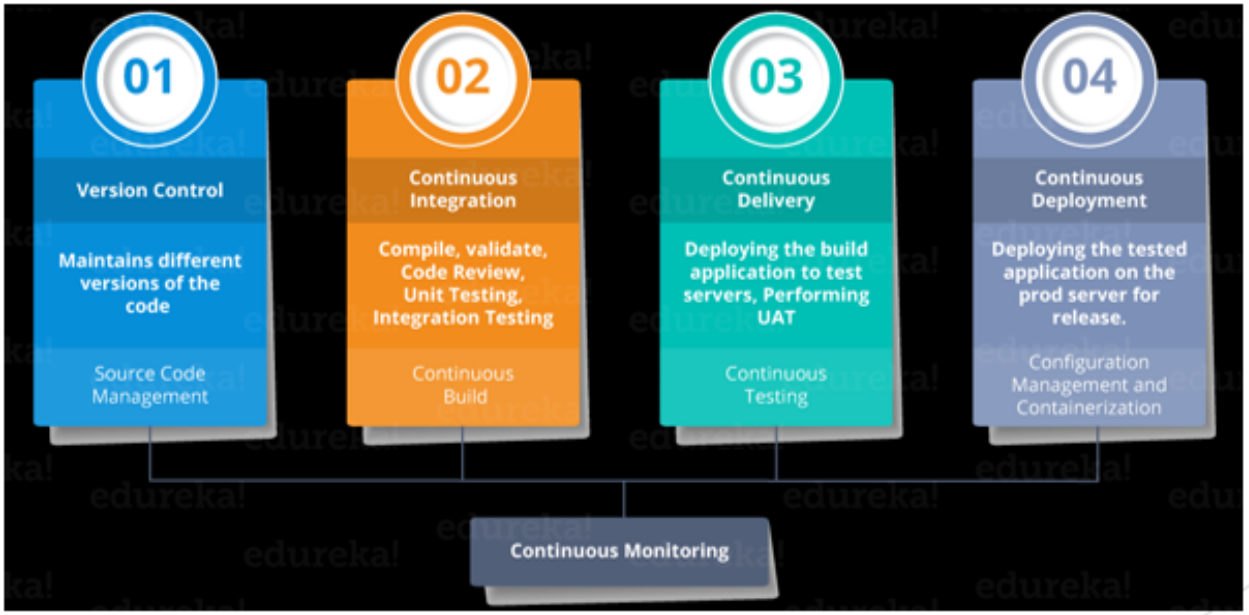


Figure 1: DevOps Lifecycle

# Continuous Integration

1. This stage is the heart of the entire DevOps life cycle.
2. It is a software development practice in which the developers require to commit changes to the source code more frequently.
3. This may be on a daily or a weekly basis. Every commit is then built and this allows early detection of problems if they are present.

1. Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging.
2. The code supporting new functionality is continuously integrated with the existing code.
3. Since there is continuous development of software, the updated code needs to be integrated continuously as well as smoothly with the systems to reflect changes to the end-users.
4. Jenkins is a very popular tool used in this phase.

# Continuous Integration

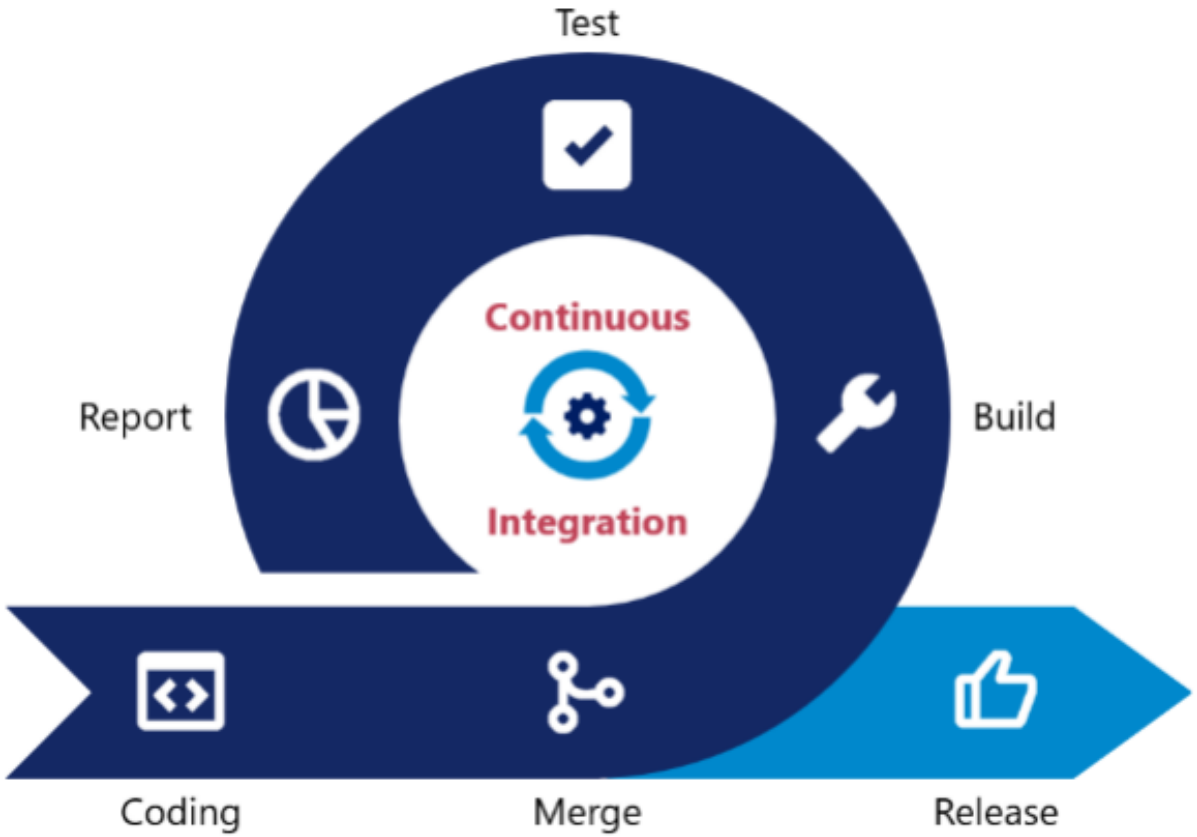


Figure 2: Continuous Integration (source: https://www.qfs.de/en/blog/article/2019/07/11/using-qf-test-in-continuous-integration-systems.html)

# Continuous Delivery

1. Continuous delivery is the important process of delivering the software/Updates to production in smaller increments, ensuring that the software can be released at any time.
2. continuous delivery is a pipeline or a life cycle of a code, where the code newly developed or updated by the software team, gets tested at different stages both through manual and automated tests and passes both the manual and automated stage gates and gets into production.

# CI Vs CD Vs CDep

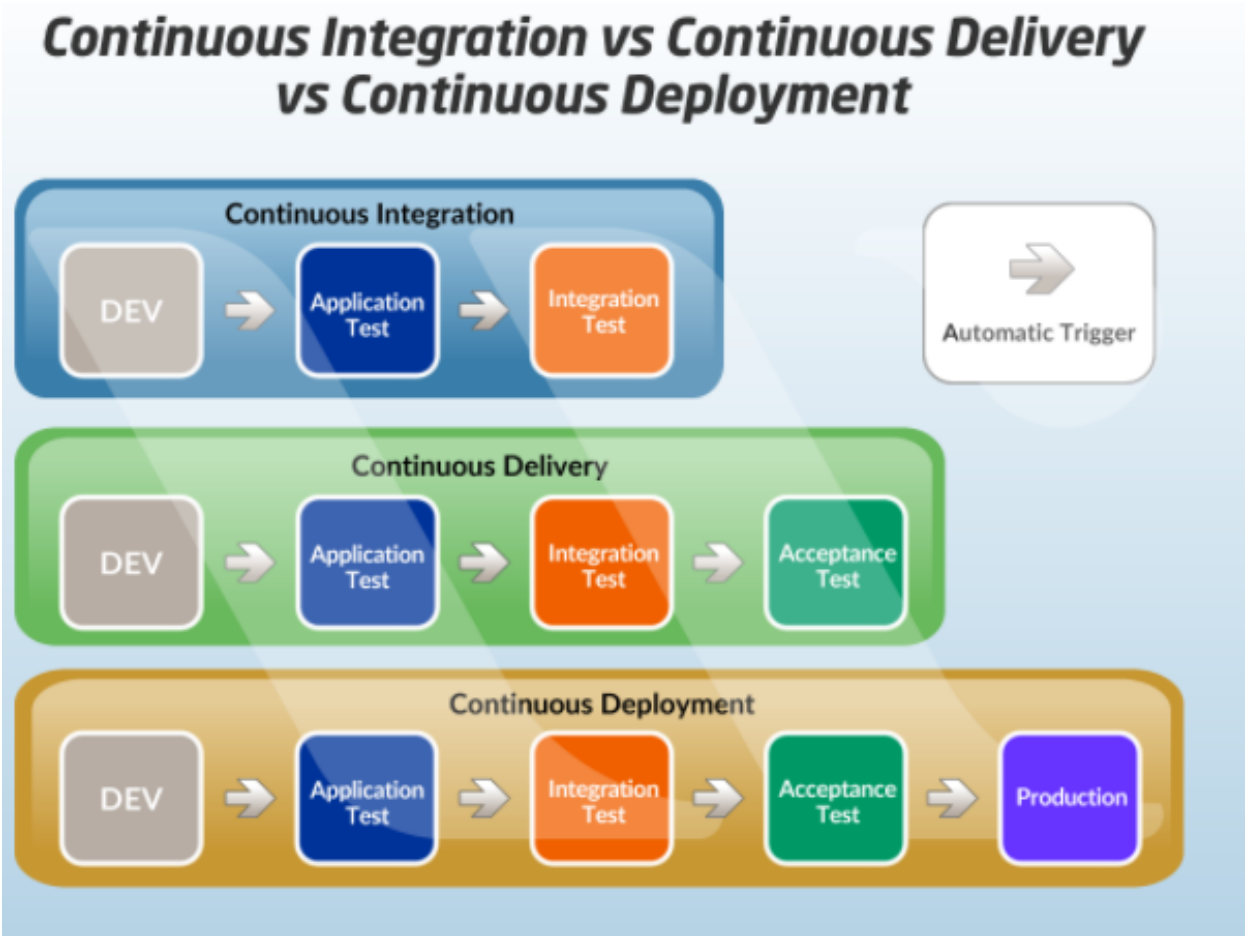


Figure 3: CI Vs CD Vs CDep (source:https://www.whizlabs.com/blog/continuous-integration-vs-continuous-delivery-vs-continuous-deployment/)

# CICD Using Jenkins

1. Jenkins is an open-source implementation of a Continuous Integration server written in Java.
2. It works with multiple programming languages and can run on various platforms (Windows, Linux, and macOS). It is widely used as a CI (Continuous Integration)  CD (Continuous Delivery) tool.
3. Many plugins available for integrating it with other tools like Slack, GitHub, Docker.
4. Jenkins supports the entire software development life cycle that includes building, testing, documenting the software and deploying.

# Jenkins Overview

Jenkins is an open-source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. It is a server-based system and can run tools such as Ant, Maven, arbitrary shell scripts and Windows batch commands.

# Jenkins Benefits

1. **Reduced development cycle** – Since every commit is getting built and tested, it allows releasing new features to the user faster and with fewer errors
2. **Time Saving:** Earlier integration of code was done manually, but the sue of Jenkins made it fast by automating the process.
3. **Automated workflow** – Teams do not have to worry about running a manual test for each commit.
4. Jenkins supports the entire software development life cycle that includes building, testing, documenting the software and deploying.

# Jenkins Pipeline

1. Jenkins Pipeline is a suite of plugins that supports implementing and integrating continuous delivery pipelines into Jenkins.
2. A Pipeline can be created in one of the following ways:
   - Through Blue Ocean - after setting up a Pipeline project in Blue Ocean, the Blue Ocean UI helps you write your Pipeline's Jenkins file and commit it to source control.
   - Through the classic UI - you can enter a basic Pipeline directly in Jenkins through the classic UI.
   - In SCM - you can write a Jenkins file manually, which you can commit to your project's source control repository.

# Built-In Documentation

1. Pipeline ships with built-in documentation features to make it easier to create Pipelines of varying complexities. This built-in documentation is automatically generated and updated based on the plugins installed in the Jenkins instance.
2.