CI-CD Fundamentals

Overview

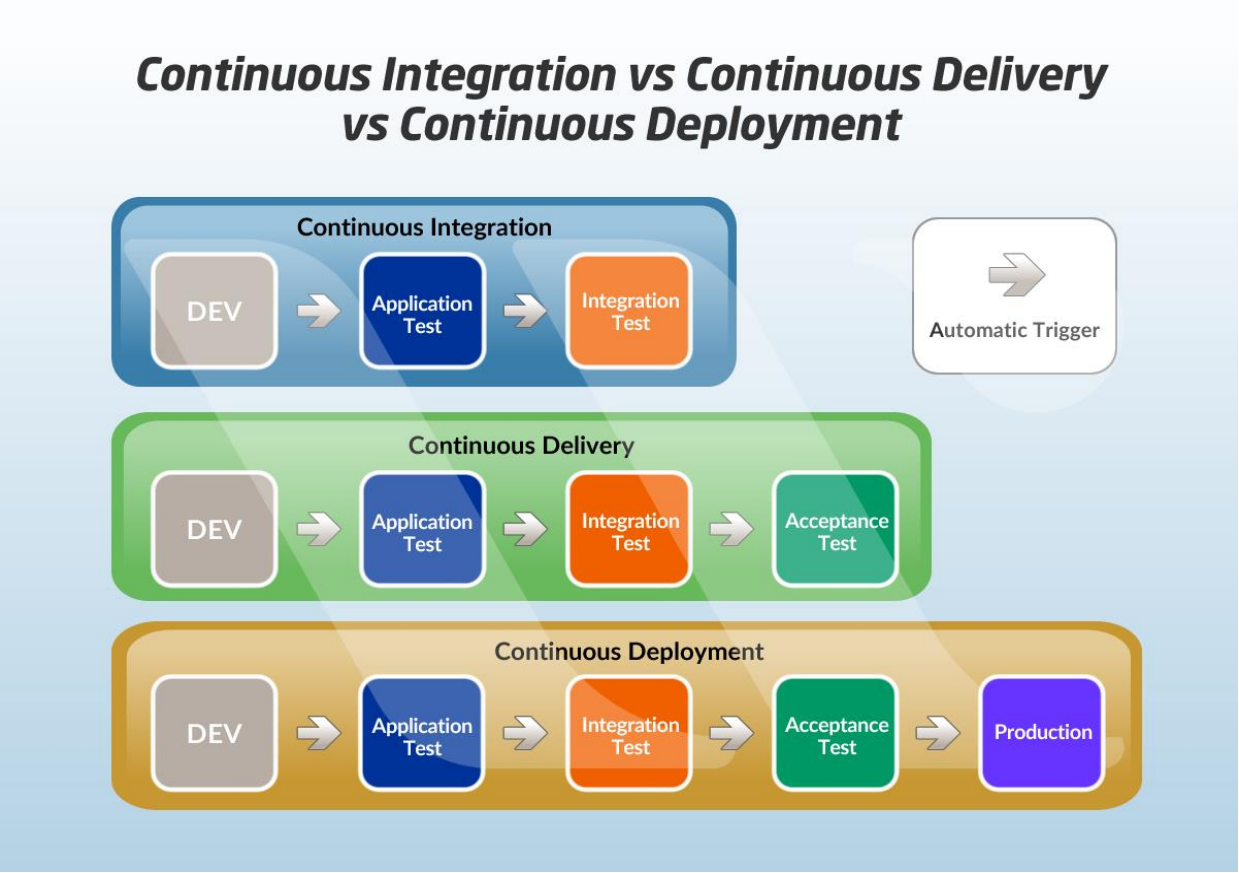
- CI/CD introduction
- CI/CD Pipeline
- Business Benefits
- Challenges
- Solution

CI/CD Introduction

CI/CD is a best practice for DevOps teams to implement, for more frequent and reliable delivery of code changes

CI/CD consist of three major aspects

- Continuous Integration
- Continuous Deployment
- Continuous Delivery



Continuous Integration

Continuous Integration is the process of merging developer branches to the main branch several times a day. CI puts an emphasis on test automation and finally generates a high quality, deployable artifact.

Continuous Deployment

Continuous Deployment is extension of Continuous Delivery in such a way that it allows frequent automated deployments without any human interaction.
Typical phases in Continuous Deployment are Infrastructure Provisioning, Smoke Testing,
Production Deployments and Automated Rollbacks.

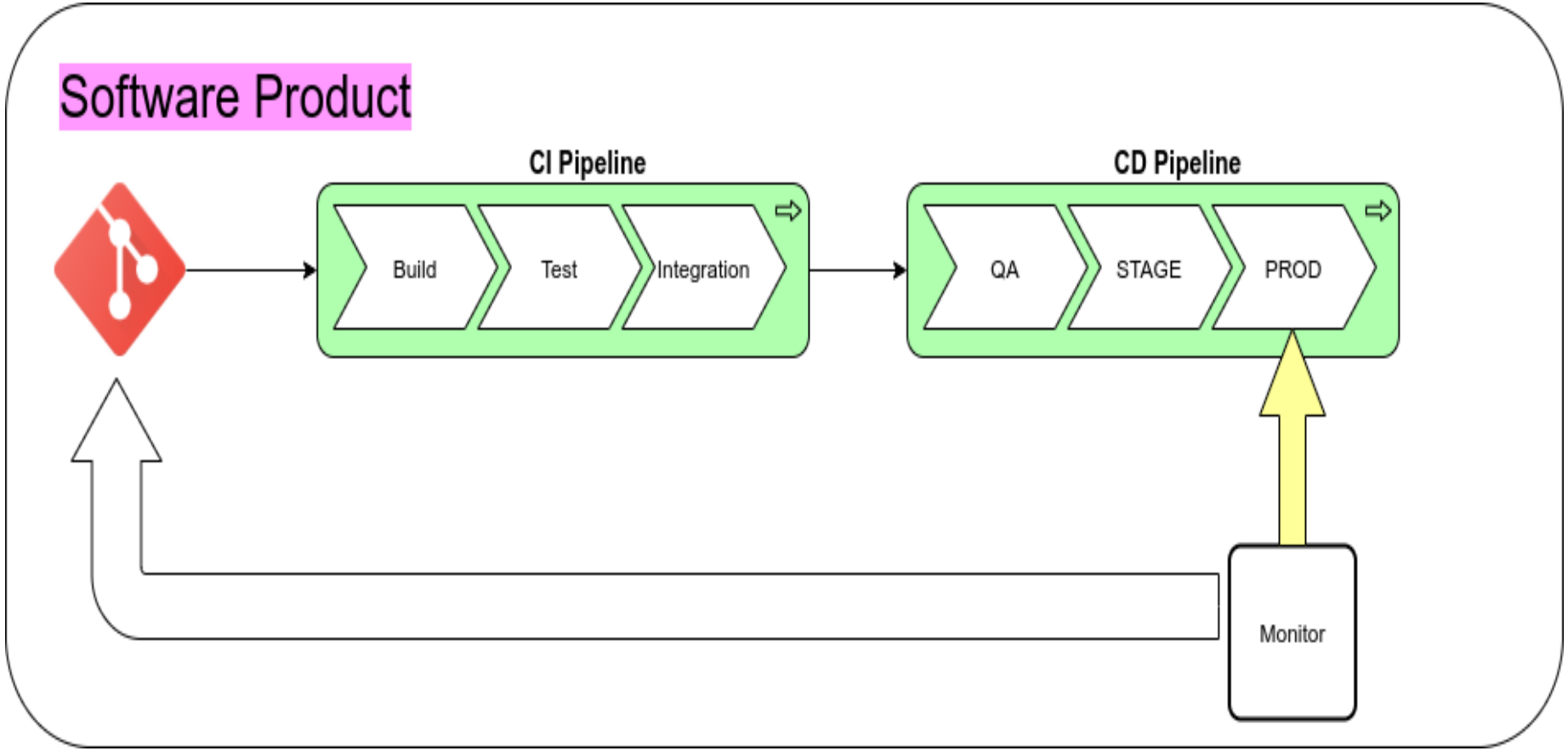
Continuous Delivery

In addition to Continuous Integration, Continuous Delivery makes sure that at any point in time changes of a software product can be released in an automated way quickly to customers

CI/CD Pipeline

CI/CD pipeline is combination of steps to be completed to deliver a new release. The CI/CD pipeline consists following stages:

- ***Build***
- ***Test***
- ***Analyze***
- ***Deploy***
- ***Verify***
- ***Promote***



Business Benefits

- **CICD** is able to catch unit test. Hence less bugs in production and less time in testing. This Avoids Cost.
- **CICD** is able to catch compile errors after merge. Hence less developer time on issues from new developed code. This Reduces cost.
- **CICD** is able to detect security vulnerabilities. This prevents embarrassing or costly security loopholes and Avoids cost.
- **CICD** is able to automate infrastructure creation leading to less human error and faster deployments. This Saves money.
- **CICD** is able to automate infrastructure cleanup leading to less infrastructure costs from unused resources. This Saves Money by reducing extra infrastructure costs.
- **CICD** gives us faster and more frequent production deployments and new value-generating features are released more quickly. Hence, Increases Revenue

Challenges

- Delays in production deployments due to release process being manual and error-prone, affects release time lines.
- Poor Release quality, no code consistency.
- No proper backout or rollback mechanisms
- Complex Deployments – dependency on few experts
- Manual Infrastructure creation and clean up
- Inconsistent environment configurations
- Conflicting goals between development and operation teams

Solution

- CI/CD code changes are simpler/smaller and deployments are automated and have fewer unintended consequences.
- Fault isolation is easy and quick. Detecting the root cause of a fault and then pointing out the exact location of the fault is one of the most proclaimed benefits of CI/CD.
- A CI/CD pipeline enables developers to integrate their code into a common repository in smaller batches. Through this repository, developers can share their builds with the entire team rather than working in isolation. Now, the whole team can collaborate for thorough detection and fixation of the most severe bugs and also the roll back mechanism is well defined

- Using CI/CD, you can improve test reliability to a great extent. Since specific and atomic changes are introduced to the system, it allows developers or QAs to add more relevant positive and negative tests for the changes. This testing is also referred to as ‘Continuous Reliability’ within a CI/CD pipeline.
- CI/CD improves overall communication and accountability between team members. It does so by becoming a common framework for all developers, QAs, and product managers working on a particular project.
- CICD is able to automate infrastructure creation leading to no human error or Inconsistent environment configurations and faster deployments.

Thank you!