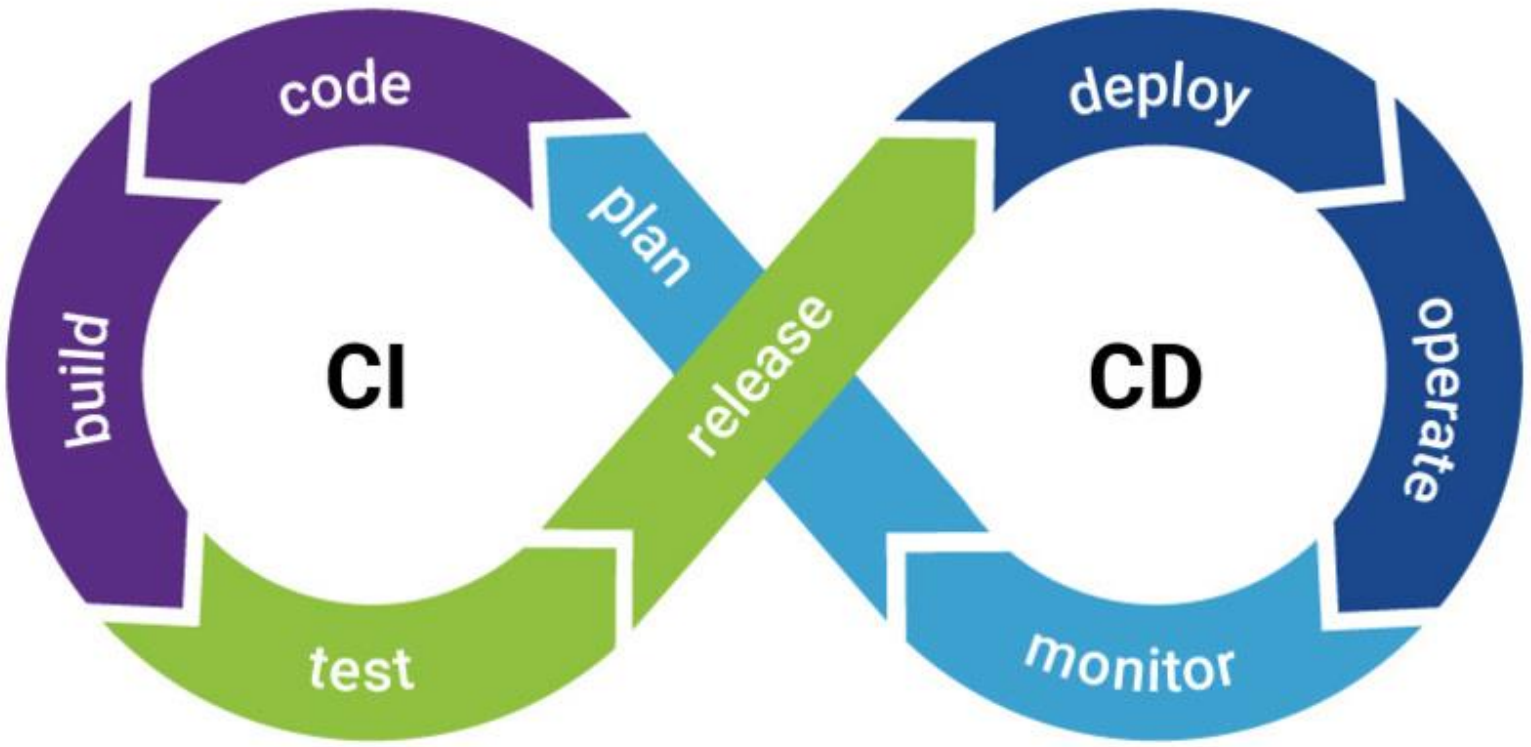


CICD Fundamentals & Business Benefits

Overview

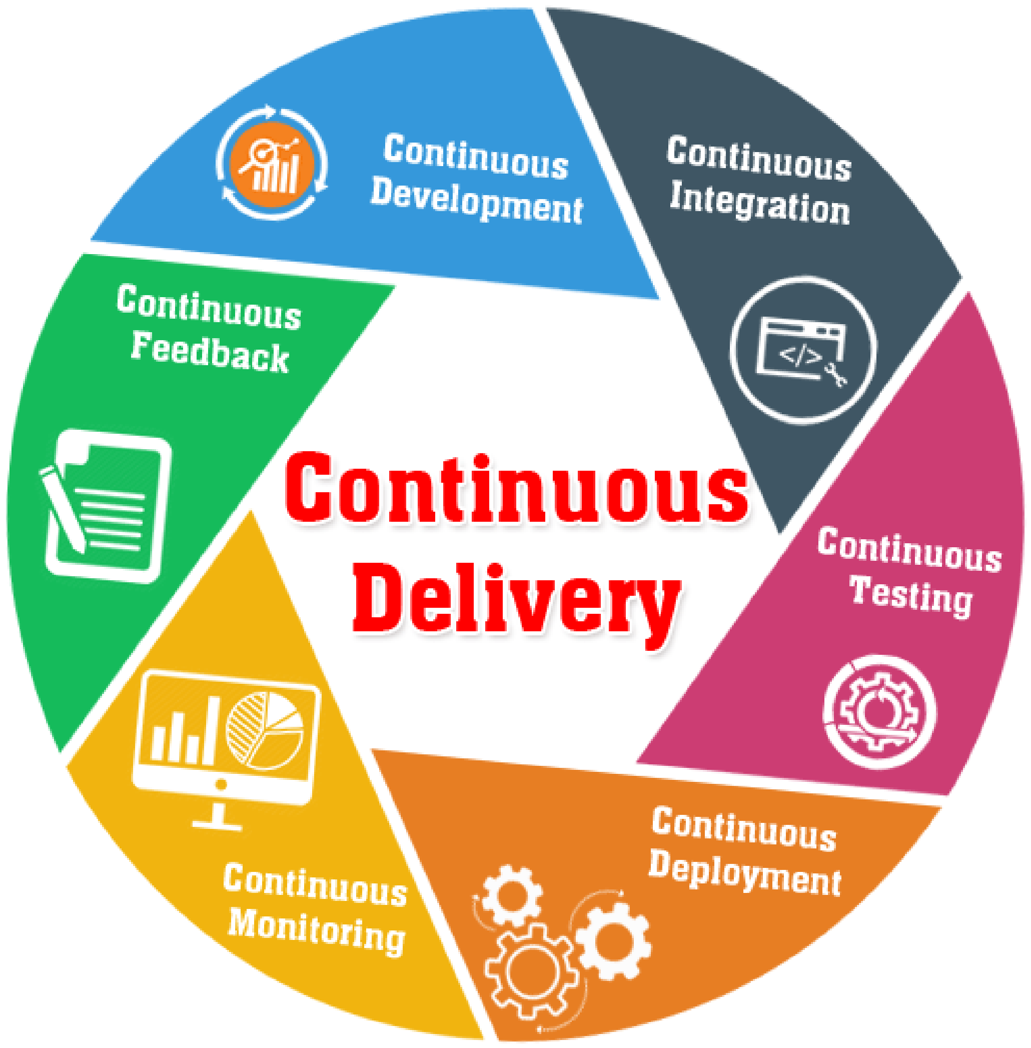
- What is CI/CD
- CI/CD pipeline
- What are the business benefits of CICD
- Current Pain Points
- CICD Solution

What is CICD

The CI/CD is one of the best practices for DevOps teams to implement, for delivering code changes more frequently and reliably

CI/CD consist of three major concepts

- Continuous Integration
- Continuous Deployment
- Continuous Delivery



Continuous Integration

Continuous Integration describes 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 extends 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

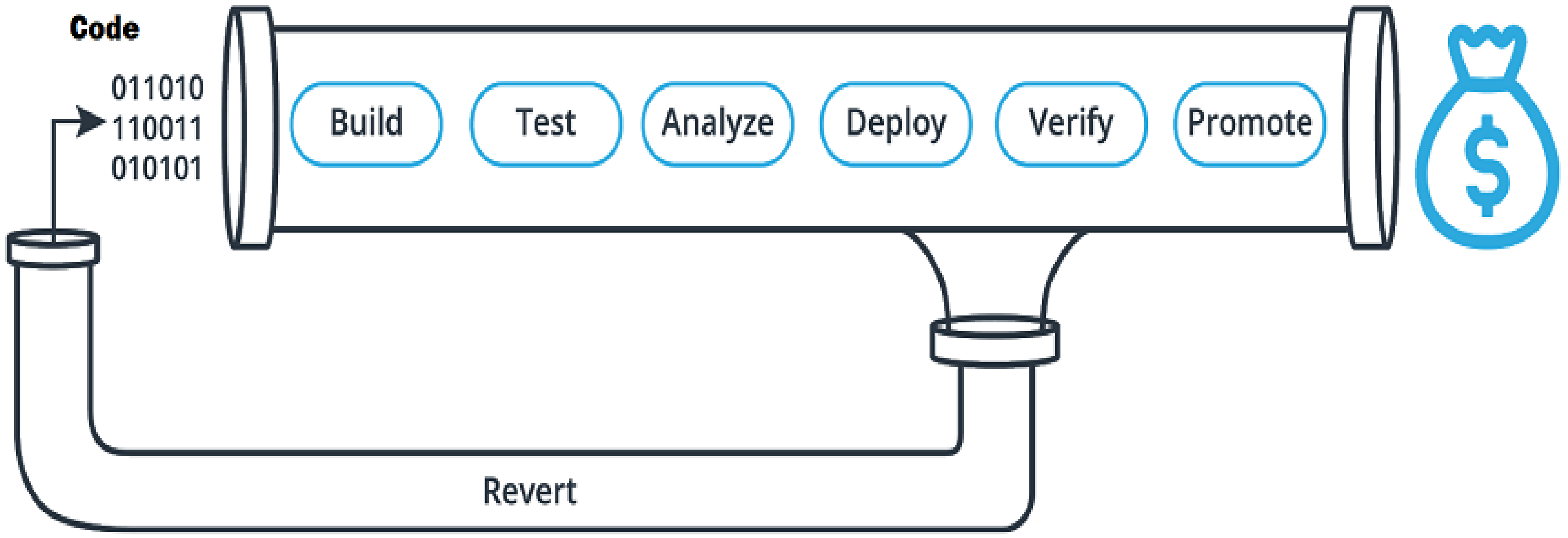
In addition to Continuous Integration, Continuous Delivery makes sure that changes of a software product can be released quickly to customers in an automated way and at any point in time.

CI/CD Pipeline

CI/CD pipelines are a series of steps that must be completed to deliver a new release.

The CI/CD pipeline typically breaks down into the following stages:

- ➢ Build
- ➢ Test
- ➢ Analyze
- ➢ Deploy
- ➢ Verify
- ➢ Promote

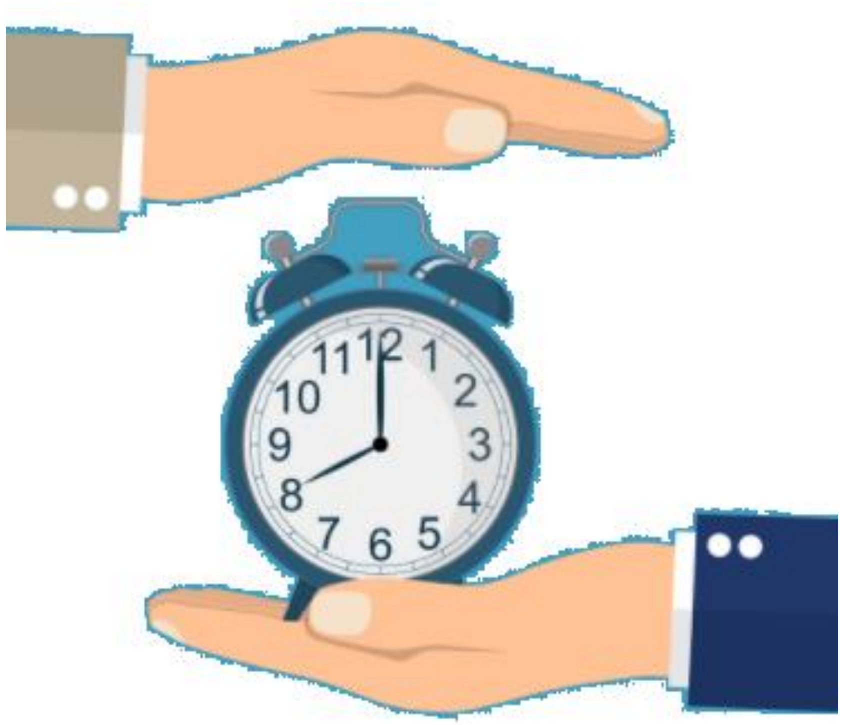


What are the Business Benefits of CI/CD?

- 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 developer code.** This **Reduces cost.**

- CICD is able to **detect security vulnerabilities.** This prevent embarrassing or costly security holes 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, **Increase Revenue**

- CICD allows us to **deploy in production without manual checks.** Features will take **less time to go to market.** Hence, **Increase Revenue**
- CICD can perform **automated smoke tests.** we would have **reduced downtime** from a deploy-related crash or major bug. Hence, **Protecting Revenue**
- CICD is able to **automate rollback triggered by job failure.** we could have quick undo to return production to working state. Hence, **Protecting Revenue**



Our Current Pain Points

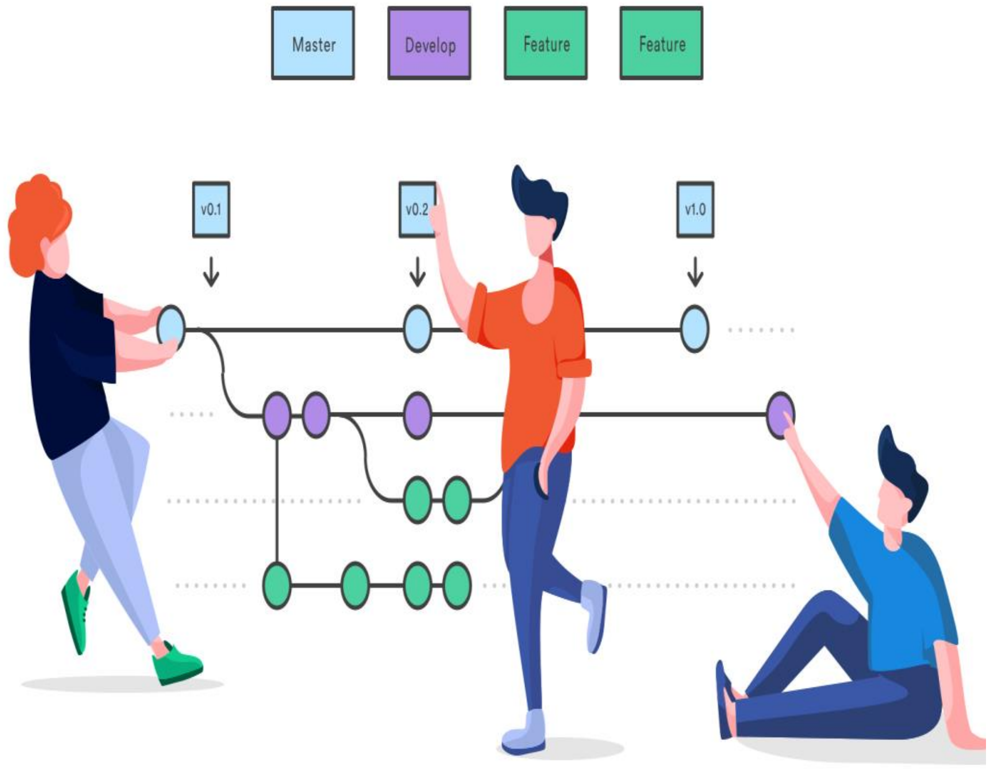
- Our release process is manual and error-prone this always leads to delays of production deployments and affects release time lines.
- Poor Release quality, no code consistency and no proper backout or rollback mechanisms
- Complex deployments. Only a chosen few experts are able to understand the whole process and tons of hand crafted helper scripts.
- Infrastructure creation and clean up is manual
- Inconsistent environment configurations
- Conflicting goals between dev and ops

CI/CD Solution

Smaller code changes are simpler and deployments are automated and have fewer unintended consequences. Fault isolation is simpler and quicker. 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. Fault isolation refers to system design, limiting the negative results of an error by pinpointing the cause and location.

A CI/CD pipeline enables developers to integrate their code into a common repository in small 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.

End-user involvement and feedback during continuous development leads to usability improvements. You can add new requirements based on customer's needs on a daily basis.