CI/CD BENEFITS PROPOSALS

KUSUMA SHIVAPURA SOMASHEKAR

OVERVIEW

- HOW PRESENT CYCLE IS WORKING?
- DRAWBACKS FROM THE CURRENT WORK.
- INTRODUCING NEW CYCLE.
- BENEFITS FROM NEW CYCLE.

- HOW PRESENT CYCLE IS WORKING?

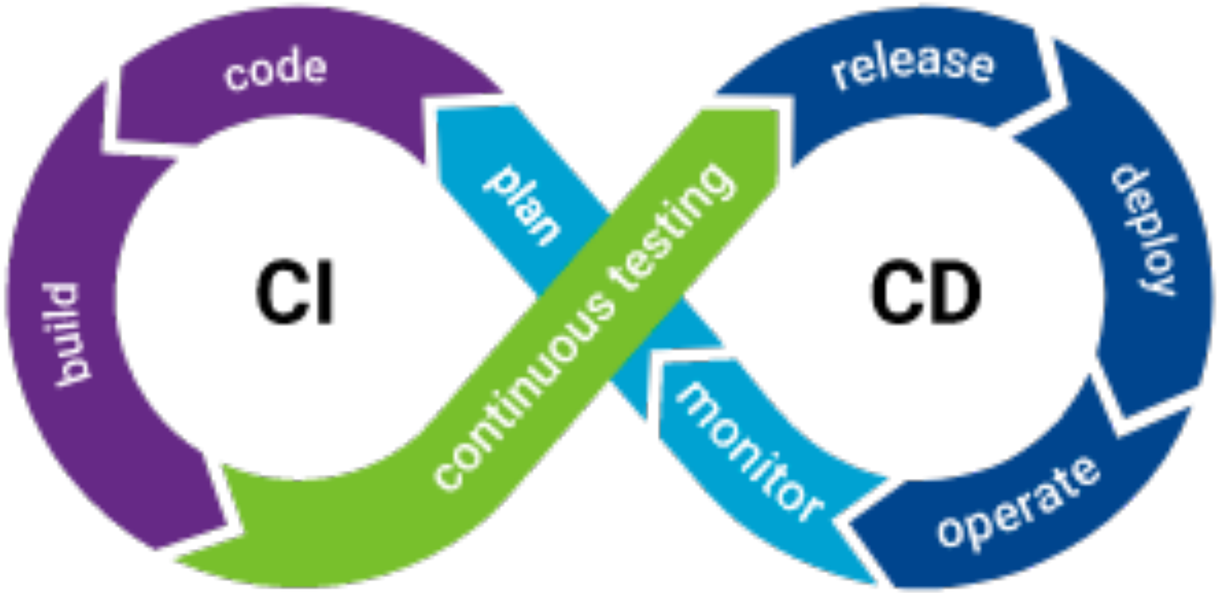
- The release cycle began with change management tickets, which each team was required to file eight days before the deploy.
- Product owners and development teams totalling about 30 people would meet to discuss the current set of releases every Wednesday at a CM coordination meeting.

Drawback:

- Investing more time in a release cycle than delivering value
- Going through integration hell every time we finish a feature
- Code gets lost because of botched merges
- Unit test suite hasn't been green in ages
- Deployments contribute to schedule slip
- Friction between ops and development departments
- Only one engineer can deploy a system
- Deployments are not cause for celebration

- INTRODUCING NEW CYCLE.

- continuous Integration
- continuous delivery
- continuous deployment



Continuous Integration

- The practice of merging all developers' working copies to a shared mainline several times a day. It's the process of "Making". Everything related to the code fits here, and it all culminates in the ultimate goal of CI: a high quality, deployable artifact!
- Some common CI-related phases might include:

- Compile
- Unit Test
- Static Analysis
- Dependency vulnerability testing
- Store artifact
- Continuous Deployment

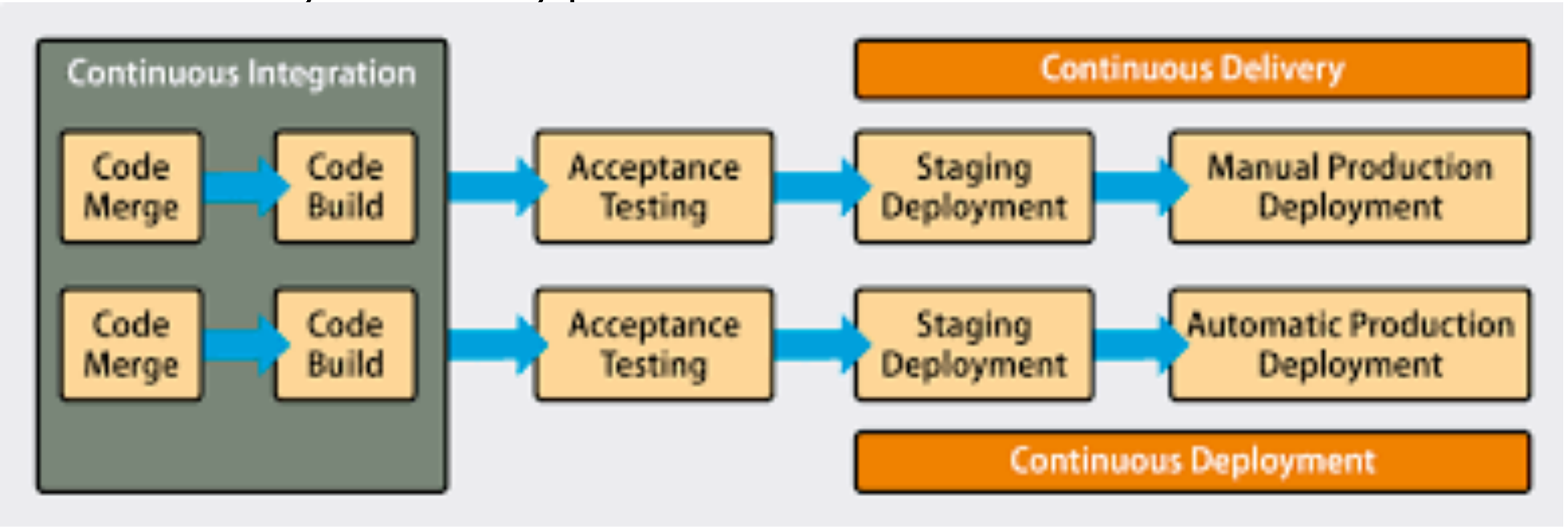
continuous deployment

- A software engineering approach in which the value is delivered frequently through automated deployments. Everything related to deploying the artifact fits here. It's the process of "Moving" the artifact from the shelf to the spotlight.
- Some common CD-related phases might include:

- Creating infrastructure
- Provisioning servers
- Copying files
- Promoting to production
- Smoke Testing (aka Verify)
- Rollbacks

continuous delivery

- In addition to continuous integration , continuous Delivery makes sure that changes of a software product can be release quickly to customers in an automated way and at any point in time.



CI/CD TO THE RESCUS.
HOW WE COULD BENEFIT FROM DEVOPS PRINCIPLES

Problem: "We need to ensure compliance with our off-shore team"

A SaaS client prepared for launch. An off-shore team developed their app. The off-shore team was not allowed to have access to the production environment for security and compliance purposes. So, the client needed the off-shore team to deploy the infrastructure without doing manual provisioning in the console.

Solution: A CI/CD pipeline that links to production account

We worked with the SaaS start-up to implement an Infrastructure as Code (IAC) methodology and defined the infrastructure in CloudFormation templates. We then built a CI/CD pipeline which allows the off-shore team to deploy to the production account. This works via a ‘commit’ to source control, and the CI/CD then deploys to the production account.

Business Impact: Off-shore resources reduce cost and meet compliance

The SaaS client met their launch deadline. The off-shore team was set-up to contribute within compliance standards. The off-shore team continued to build and deploy infrastructure via CI/CD pipelines. This implementation increased speed to deployment, satisfied compliance needs, and reduced operational overhead by 30% using off-shore resources.

BENEFITS

• Reduce Cost	Less developer time on issues from new developer code
• Avoid Cost	Less bugs in production and less time in testing
• Avoid Cost	Prevent embarrassing or costly security holes
• Avoid Cost	Less human error, Faster deployments
• Reduce Cost	Less infrastructure costs from unused resources
• Increse Revenue	New value-generating features released more quickly
• Increse Revenue	Less time to market
• Protect Revenue	Reduced downtime from a deploy-related crash or major bug
• Protect Revenue	Quick undo to return production to working state

THANK YOU