

ADOPTING CI/CD FOR UDAPEOPLE APP

Achieve, Build and Deploy Automation

Presented by:

Elijah Ayandotun Aremu

Technical Support Engineer



CI/CD : 3 Major Concepts

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 Delivery

Asides from 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 time when due or requested. Continuous Deployment ensures that value is delivered frequently through automated deployments.

Continuous Deployment

Continuous Deployment extends Continuous Delivery as it allows frequent automated deployments without any human interaction.

Phases in Continuous Deployment are Infrastructure Provisioning, Smoke Testing, Production Deployments and automated Rollbacks.

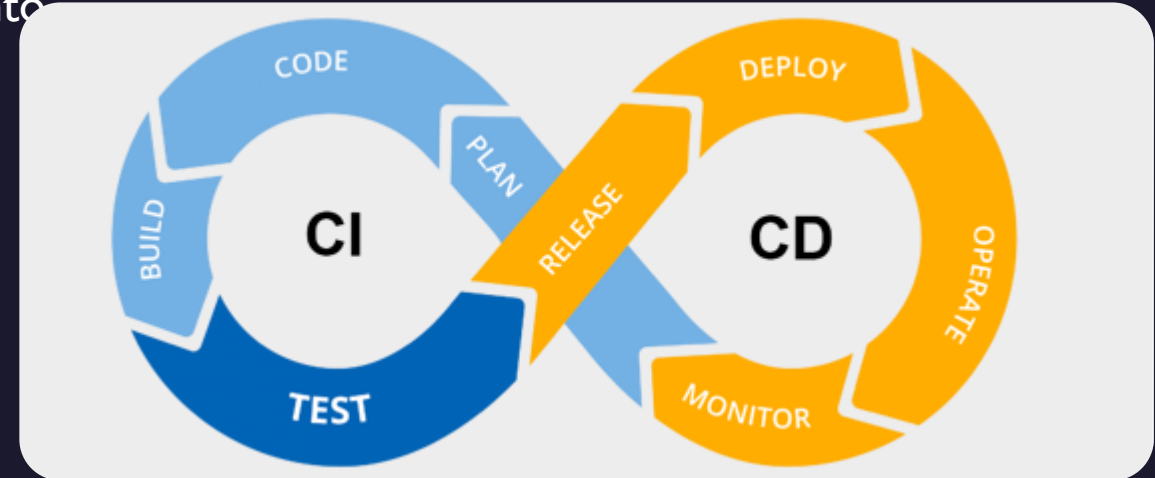
CONTINUOUS INTEGRATION

CONTINUOUS INTEGRATION - CI

This is the practice of merging all working copies of a project into a shared main/master copy, several times a day.

A typical pipeline for Continuous Integration includes:

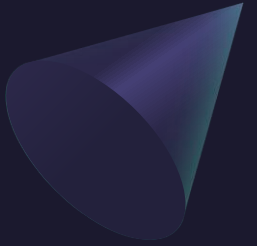
- Detect changes in the source code repository (new commits appear)
- Source code quality analysis
- Build
- Execute all unit tests
- Execute all integration tests
- Generate deployable artifacts
- Report status



If one of the steps above fails:

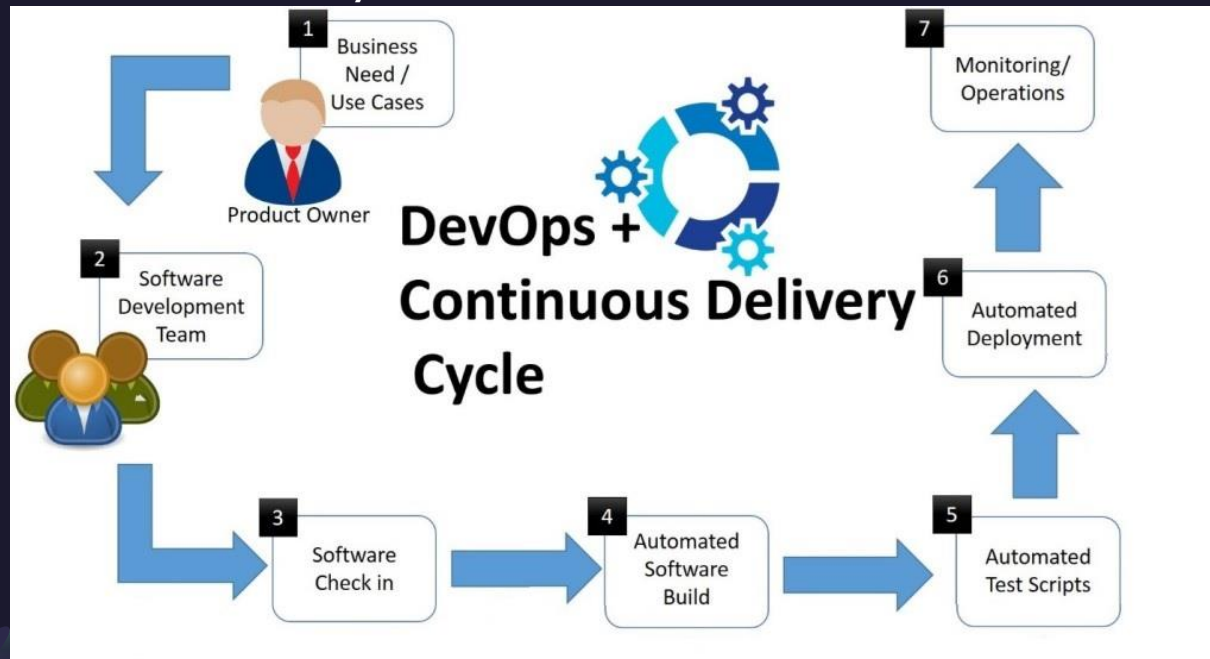
- Integration may stop or continue depending on defect severity and configuration
- Results are notified to the team via email or chat system
- The team fixes defects and commits again
- Tasks are performed again

CONTINUOUS DEPLOYMENT



CONTINUOUS DELIVERY

makes it possible to release builds to the production environment when needed. Allowing the team to deploy at will, CD effectively reduces time to market.



CONTINUOUS DEPLOYMENT

is an extension of continuous delivery that automatically deploys each build that passes the full test cycle.

Instead of waiting for a humans to decide what and when to deploy to production, a continuous deployment system deploys everything that has successfully traversed the deployment pipeline, from start to finish, following several testing.

Keep in mind that when new code is automatically *deployed*, new features can still be activated conditionally later or for a subset of users. Deploying automatically pushes features and fixes to customers quickly, encourages smaller changes with limited scope, and helps avoid confusion over what is currently deployed to production.

CI/CD: BENEFITS

FAST RELEASE RATE

Failures are detected faster and hence, such failures can be repaired speedily, which leads to increasing release rates. There is a continuously merging of codes in CI/CD hence continuous deployment is done to production after thorough testing, keeping the code in a release-ready state.

REDUCE COSTS

Automation in the CI/CD pipeline reduces the number of errors that can take place in the many repetitive steps of CI and CD. This frees up developer time that could be spent on product development as there aren't as many code-changes to fix down the road if the error is caught quickly. Additionally, increasing code quality with automation also increases your ROI.

Saturday, August 13, 2022

FASTER MEAN TIME TO RESOLUTION

One of the most important business risk assurances is to keep failures to a minimum and quickly recover from any failures that do happen.

CI/CD reduces the MTTR because the code changes are smaller and fault isolations are easier to detect. Application monitoring tools such as Prometheus are a great way to find and fix failures while.

FAULT ISOLATIONS

Using CI/CD implementations ensures that fault isolations are faster to detect and easier to implement.

Fault isolations includes monitoring the system, identifying when the fault occurred, and triggering its location

CI/CD: BENEFITS continued



SMALLER CODE CHANGES


CI/CD allows the integration of small pieces of code one at a time.

These code changes are easier and simpler to handle than huge chunks of code, hence have fewer issues that may need to be repaired at a later date.

CUSTOMER SATISFACTION

When the Ci/CD approach is adopted, this keeps your product up-to-date with the latest technology. This allows you to gain new customers that appreciates and selects you over any competitor.

Tools used for this app



- 1.AWS – Cloud Provider
- 2.Circle CI– Cloud Integration Pipeline
- 3.Ansible – IT Automation
- 4.Prometheus –Analytics and Metrics

Thank You

Elijah Ayandotun Aremu

dotun32@gmail.com

