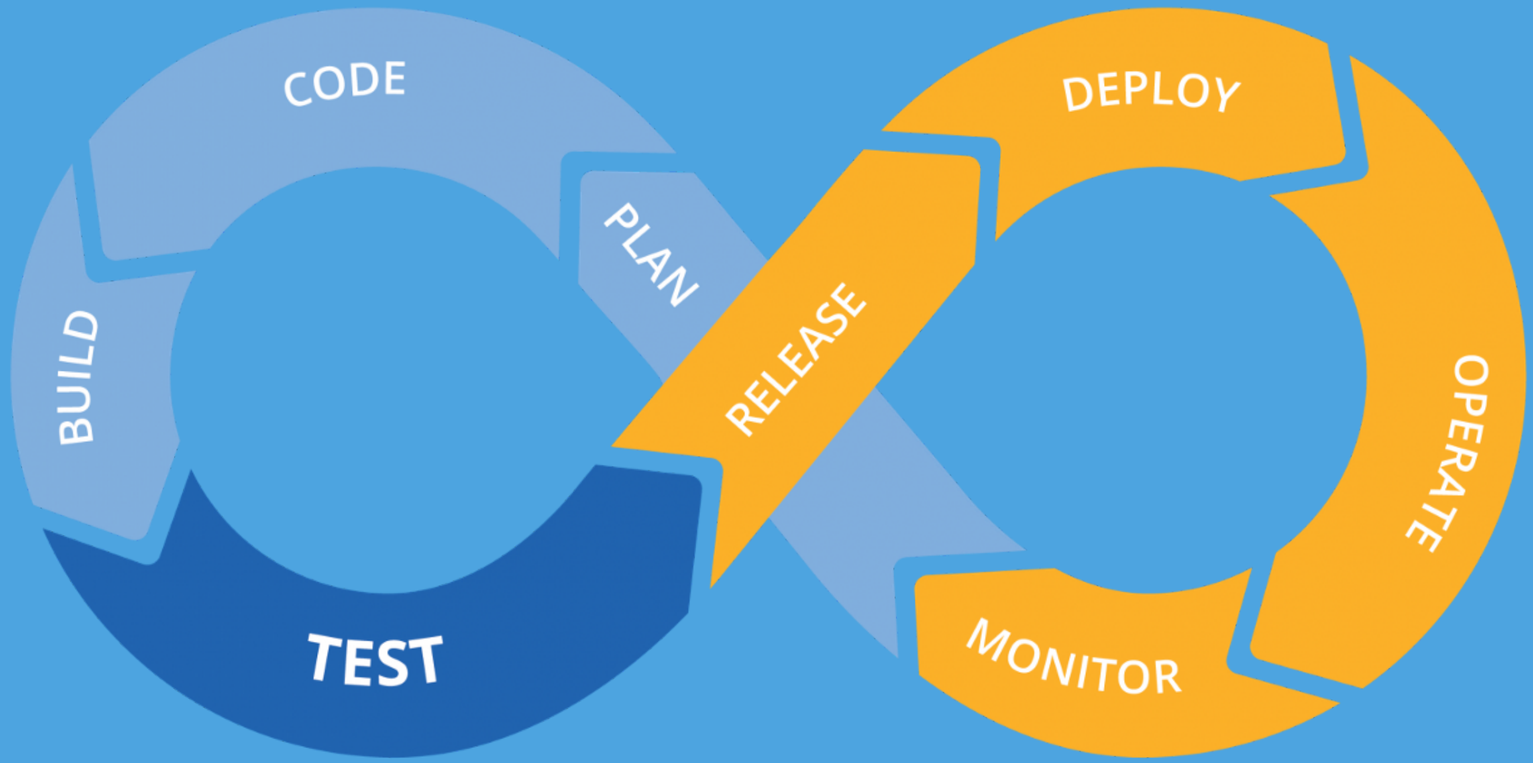
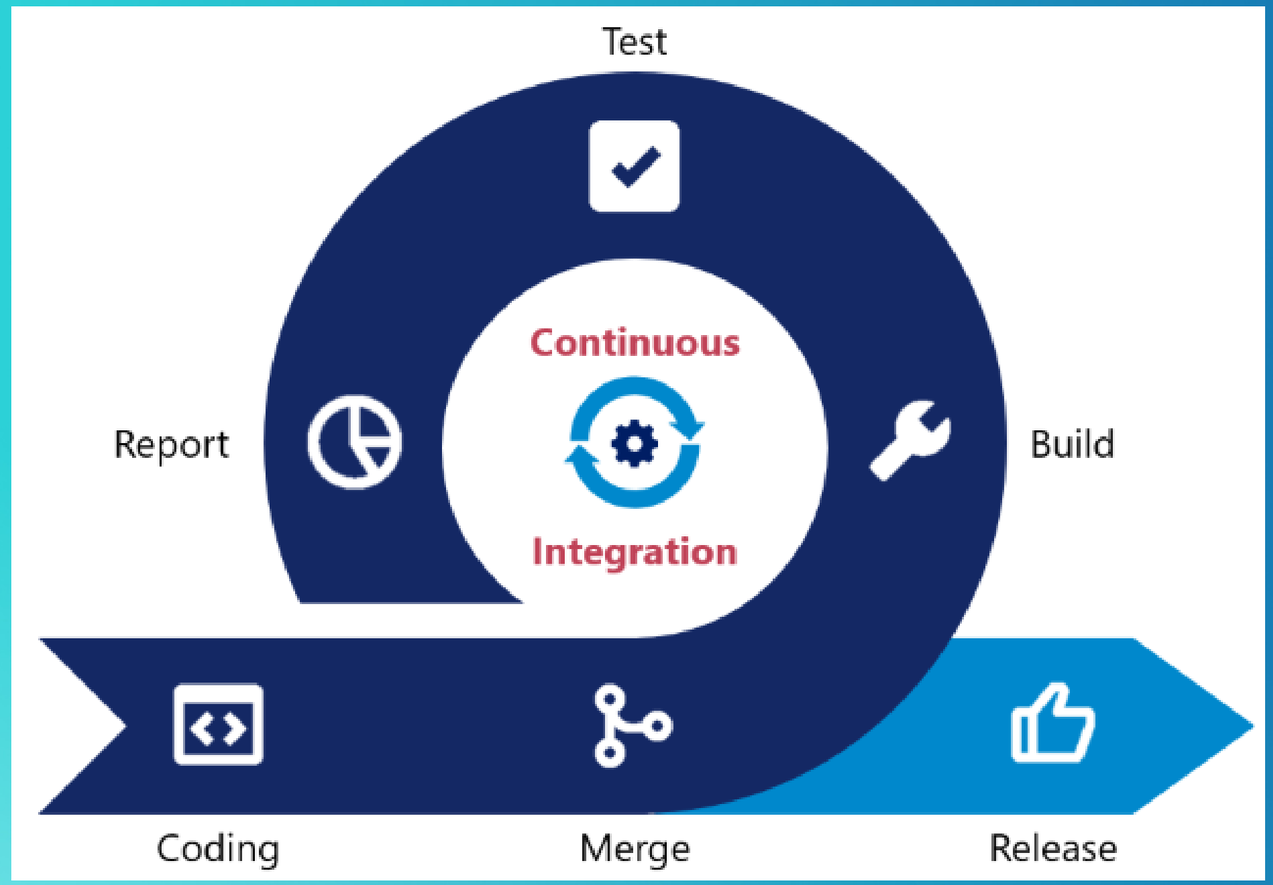


CI/CD Pipeline

A better way to build and ship your product to market

Presented by Ahmed Halaby

CONTINUOUS INTEGRATION (CI)



WHAT IS CONTINUOUS INTEGRATION ?

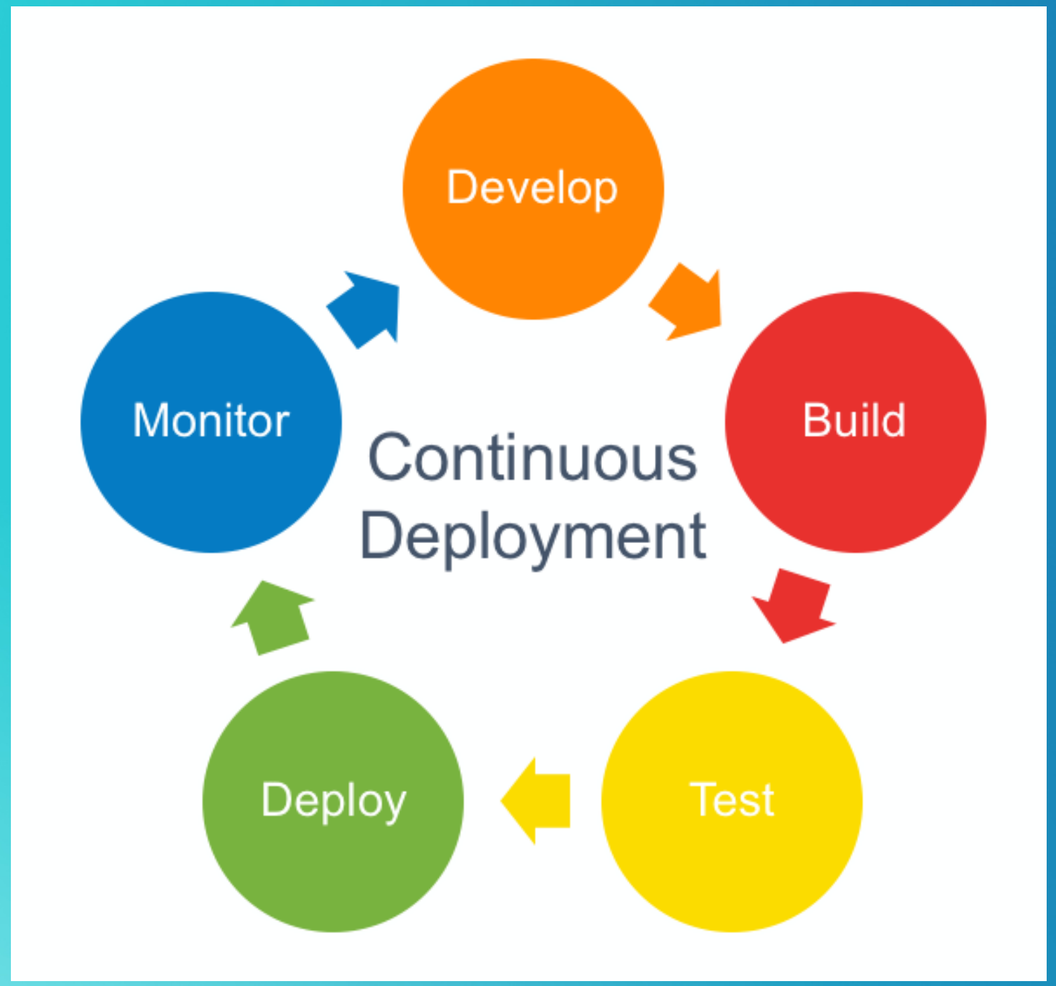
The practice of merging all developers' working copies to a shared mainline several times a day. It's the process of "Making". Everything related to the code fits here, and it all culminates in the ultimate goal of CI: a high quality, deployable artifact!

COMMON CD-RELATED PHASES

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

It's all about Code & Dev!

CONTINUOUS DEVELOPMENT (CD)



WHAT IS CONTINUOUS DEVELOPMENT?

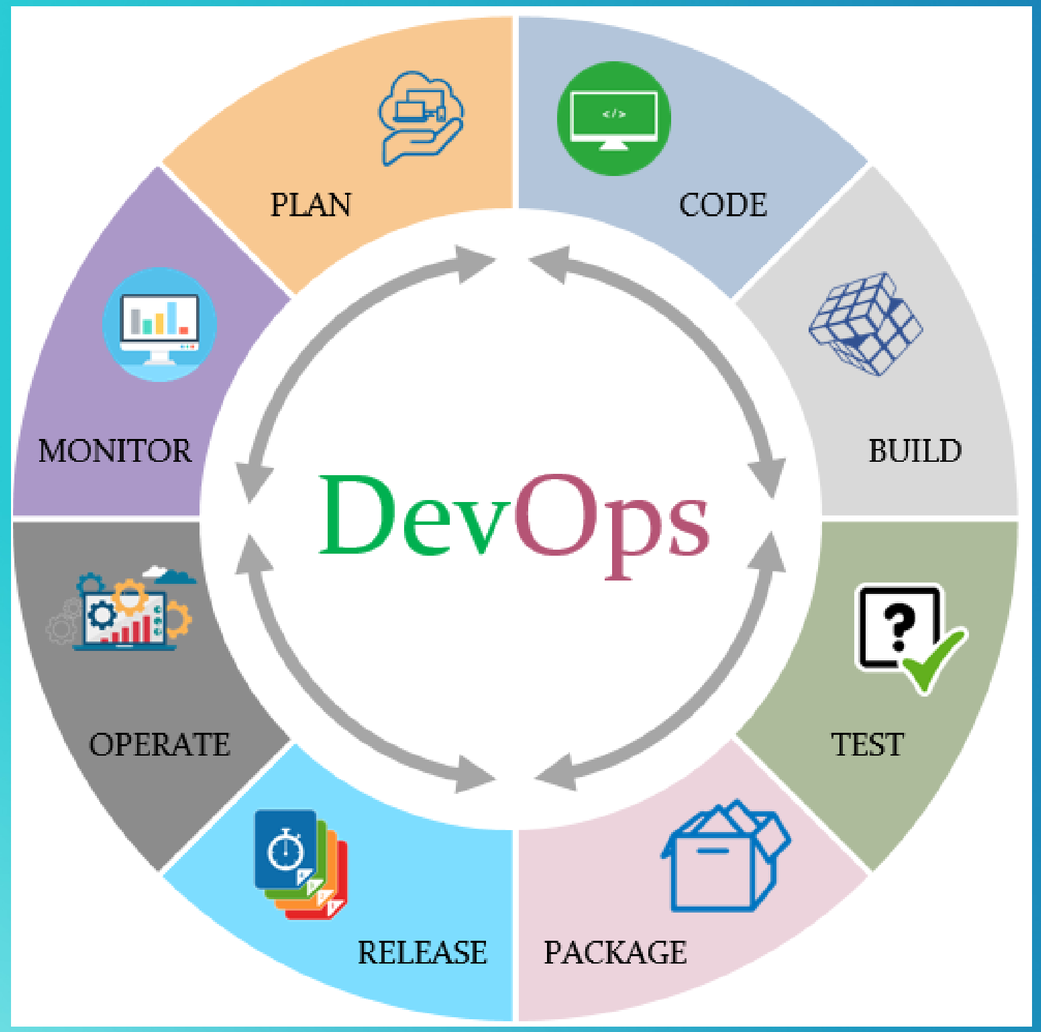
A software engineering approach in which the value is delivered frequently through automated deployments. Everything related to deploying the artifact fits here. It's the process of "Moving" the artifact from the shelf to the spotlight.

COMMON CD-RELATED PHASES

- Creating infrastructure
- Provisioning servers
- Copying files
- Promoting to production
- Smoke Testing
- Rollbacks

It's all about Ops!

CONTINUOUS INTEGRATION AND CONTINUOUS DELIVERY, DEPLOYMENT, CI/CD



WHAT IS CI/CD

Continuous integration and continuous delivery, deployment (CI/CD) count as an integral part of modern software development that is intended to decrease errors at the time of integration and development done to increase project velocity. CI/CD is actually a philosophy and a set of practices that are often augmented by robust tools that focus on automated testing at each stage of the software pipeline.

With the incorporation of CI/CD in your practice, you can reduce the time needed to integrate changes for a release and thoroughly test the change before moving into production. CI/CD comes with many advantages, but successful implementation needs a careful plan and a well thought of consideration. Choosing how to use the continuous integration tools and the changes that are needed in the environment or process can be pretty challenging without trial and error. That being said, sticking to the best practices can be helpful in avoiding common problems and attain quick improvement.

WHY SHOULD WE USE CI/CD

BENEFITS OF CI/CD

REDUCE COST

- Less developer time on issues from new developer code
- Less infrastructure costs from unused resources

AVOID COST

- Less bugs in production and less time in testing
- Prevent embarrassing or costly security holes
- Less human error, Faster deployments

INCREASE REVENUE

- New value-generating features released more quickly
- Less time to market

PROTECT REVENUE

- Reduced downtime from a deploy-related crash or major bug
- Quick undo to return production to working state