Fundamental CI/CD

Continuous Integration (CI)

*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:*

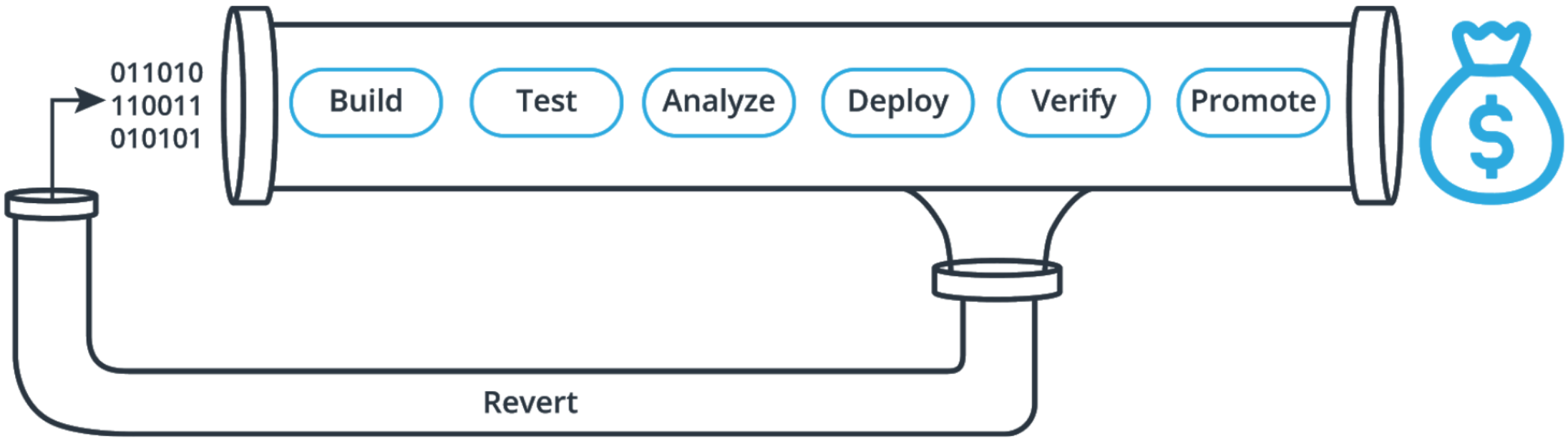
- Compile
- Unit Test
- Static Analysis
- Dependency vulnerability testing
- Store artifact

Continuous Deployment (CD)

*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

The CI/CD Pipeline



BENEFITS CI/CD

What CI/CD provide – How is it can affect – The Value we can gain from

- Catch compile errors after merge - Less developer time on issues from new developer code - Reduce Cost
- Catch unit test after failures - Less bugs in production and less time in testing - Avoid Cost
- Detect security vulnerabilities - Prevent embarrassing or costly security holes - Avoid Cost
- Automate infrastructure creation - Less human error, Faster deployments - Avoid Cost
- Automate infrastructure cleanup - Less infrastructure costs from unused resources - Reduce Cost

What CI/CD provide – How is it can affect – The Value we can gain from

- Faster and more frequent production deployment - New value-generating features released more quickly - Increase Revenue
- Deploy to production without manual check - Less time to market - Increase Revenue
- Automated smoke tests - Reduced downtime from a deploy-related crash or major bug - Protect Revenue
- Automated rollback triggered by job failure - Quick undo to return production to working state - Protect Revenue