CI/CD (Continuous Integration/ Continuous Deployment)

Road to Endless Application Development

CI (Continuous Integration)

Software developers often work in isolation, so it took days or even weeks for software developers to integrate their code and also to merge changes from different branches of code. These long periods of time created many merge conflicts, bugs, code strategy divergence and duplicated effort.

CI (Continuous Integration) requires the team's code be merged to a shared control branch (shared online storage) continuously, to avoid these problems.

Principles of Continuous Integration (CI)

- The maintenance of a code repository(shared online storage for code)
- Automating the build process
- Making the build self-testing
- Everyone committing to the baseline every day

CI-related phases might include:

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

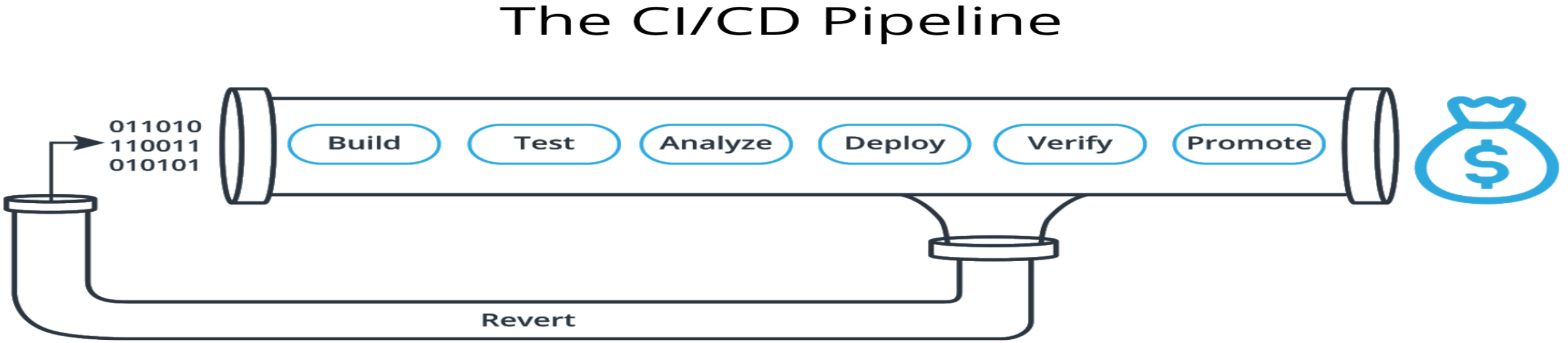
CD (Continuous Deployment)

- **CD (Continuous Deployment)**

It is all about automatically converting the application from a line of codes (Done efficiently on CI Stages) to an online application after passing with accurate tests, so that it will be used efficiently by customer and returning a revenue.

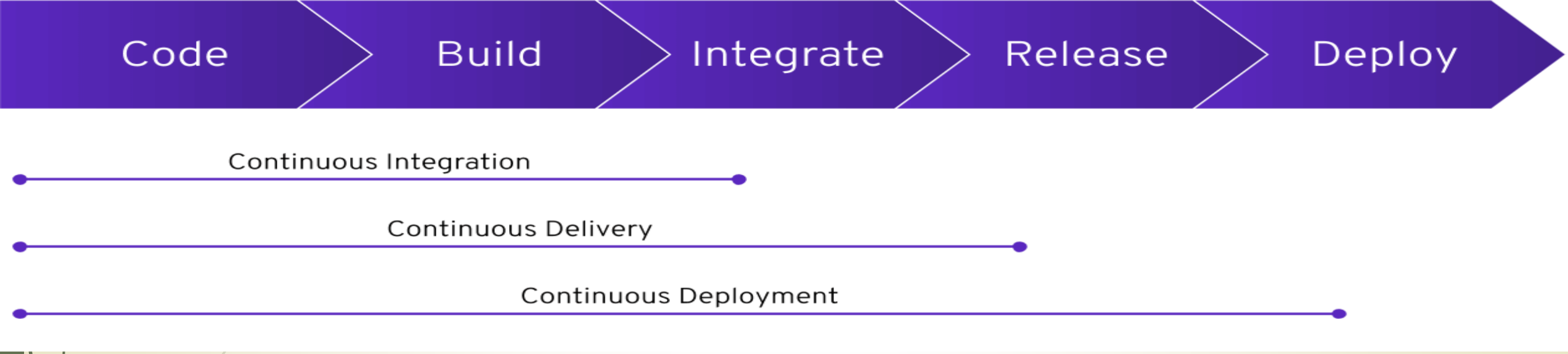
Some common CD-related phases might include:

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



Continuous Delivery is the merging of CI with CD into one CI/CD pipeline.

It gives you the ability to get changes of all types—including new features, configuration changes, bug fixes and experiments—into production, or into the hands of users, *safely* and *quickly (short cycles)* in a *sustainable* way.



Advantages of Continuous Deployment after following CI/CD pipeline

Continuous Delivery Features	Description	Advantage
Catch Compile Errors After Merge	Less developer time on issues from new developer code	Reduce Cost
Catch Unit Test 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
Faster and More Frequent Production Deployments	New value-generating features released more quickly	Increase Revenue
Deploy to Production Without Manual Checks	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