The Fundamentals and Benefits of CI/CD to Achieve, Build, and Deploy Automation for Cloud-Based Software Products

By

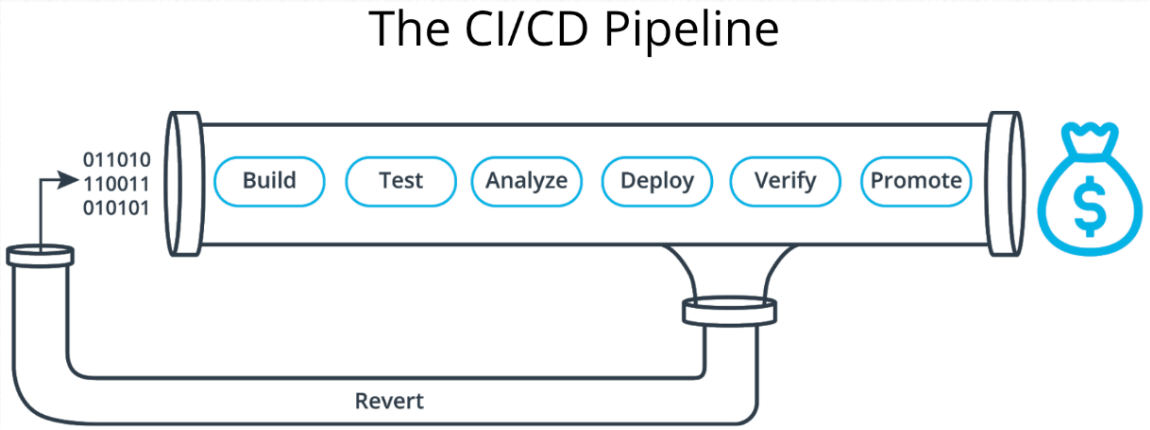
Stanley Eze

Continuous Integration (CI) is 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) is 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 Phases of CI/CD Pipeline

- Pipeline: A set of data processing elements connected in series, where the output of one element is the input of the next one.
- Continuous Integration: The practice of merging all developers' working copies to a shared mainline several times a day.
- Continuous Delivery: An engineering practice in which teams produce and release value in short cycles.
- Continuous Deployment: A software engineering approach in which the value is delivered frequently through automated deployments.
- Infrastructure as Code: The management of infrastructure using code.
- Provisioning: The process of setting up IT infrastructure.
- Artifact: A product of some process applied to the code repository.
- DevOps: A set of practices that works to automate and integrate the processes between software development and IT teams.
- Testing: A practice that seeks to ensure the quality of the software.

Benefits of CI/CD to Achieve, Build, and Deploy Automation for Cloud-Based Software Products

Continuous integration

What you need (cost)

- Your team will need to write automated tests for each new feature, improvement or bug fix.
- You need a continuous integration server that can monitor the main repository and run the tests automatically for every new commits pushed.
- Developers need to merge their changes as often as possible, at least once a day.

What you gain

- Less bugs get shipped to production as regressions are captured early by the automated tests.
- Building the release is easy as all integration issues have been solved early.
- Less context switching as developers are alerted as soon as they break the build and can work on fixing it before they move to another task.
- Testing costs are reduced drastically – your CI server can run hundreds of tests in the matter of seconds.
- Your QA team spends less time testing and can focus on significant improvements to the quality culture.

Continuous delivery

What you need (cost)

- You need a strong foundation in continuous integration and your test suite needs to cover enough of your codebase.
- Deployments need to be automated. The trigger is still manual but once a deployment is started there shouldn't be a need for human intervention.
- Your team will most likely need to embrace feature flags so that incomplete features do not affect customers in production.

What you gain

- The complexity of deploying software has been taken away. Your team doesn't have to spend days preparing for a release anymore.
- You can release more often, thus accelerating the feedback loop with your customers.
- There is much less pressure on decisions for small changes, hence encouraging iterating faster.

Continuous deployment

What you need (cost)

- Your testing culture needs to be at its best. The quality of your test suite will determine the quality of your releases.
- Your documentation process will need to keep up with the pace of deployments.
- Feature flags become an inherent part of the process of releasing significant changes to make sure you can coordinate with other departments (support, marketing, PR...).

What you gain

- You can develop faster as there's no need to pause development for releases. Deployments pipelines are triggered automatically for every change.
- Releases are less risky and easier to fix in case of problem as you deploy small batches of changes.
- Customers see a continuous stream of improvements, and quality increases every day, instead of every month, quarter or year.

One of the traditional cost associated with continuous integration is the installation and maintenance of a CI server. But you can reduce significantly the cost of adopting these practices by using a cloud service like Circle CI which adds automation to any git repository. By simply adding a configuration file at the root of your repository you will be able to create a continuous deployment pipeline that gets executed for every new change pushed to the branch.