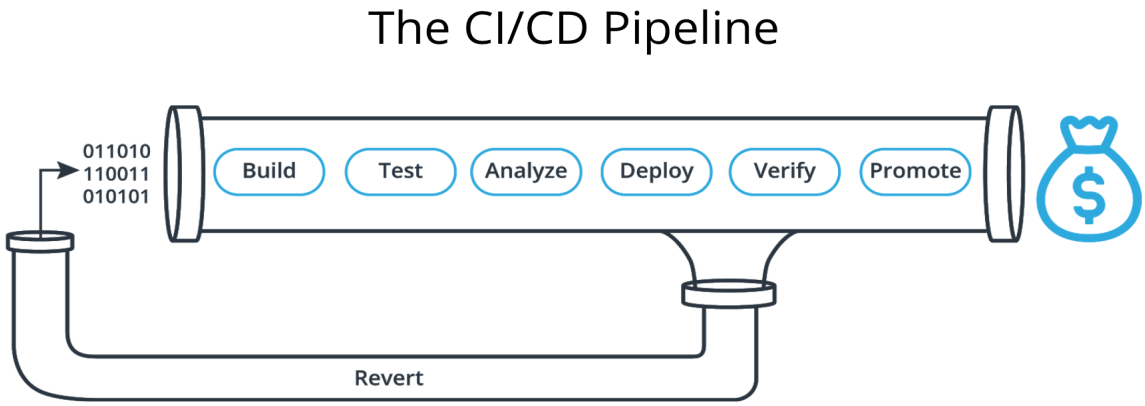
Selling CI/CD - Continuous Integration/Continuous Deployments



Premise and Objective

Understanding our organization and industry holistically is as important as gauging the value our clients gain from our software products. I cannot emphasize enough that the bulk of our reputation as an organization comes from developing quality software products that reflect what our real world would look like. Dealing genuinely good high-end products and promising short software release life-cycles for the same, and apt resolution mechanisms for issues is core in this engagement if we were to meet the objectives of demand and supply. The case is - our current software release practice isn't scaling well to this demand. That means we consume a lot of effort on manual repetitive tasks, integration, we spend less time delivering value, and issues to do with botched deployments, to mention a few. It's a huge list.

Jany Muong
UdaPeople Product Team Lead & Cloud DevOps Engineer

Premise

What CI/CD IS

With ***Continuous Integration - CI***, we're adopting a better way to build and ship products, merging all developers' working copies to a shared mainline several times a day. It's the process of "**Making**". This involves working with source control and it all culminates in the ultimate goal of CI: a high quality, deployable artifact!

Sample workflow: Compile, Unit Test, Static Analysis, Dependency vulnerability testing, Store artifact

Continuous Deployment - CD entails engineering a systematic 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

Sample workflow: Creating infrastructure, Provisioning servers. Copying files, Promoting to production, Smoke Testing (aka Verify), and Rollbacks.

I want to propose that we adopt the use of Continuous Integration and Continuous Delivery/Deployment (CI/CD) in our software development lifecycle. This will enable us to minimize cost or avoiding it entirely, and protect and/or improve revenue.

Benefits of CI/CD

~~avoiding~~ Cost
reducing Cost
increasing Revenue
~~protecting~~ Revenue

Based on this information and prior experiences with manual workflows, a CI/CD workflow is promising. Some of the key benefits are listed out below:

- Automated Infrastructure Creation: provisioning resources but with less human error, which means faster deployments and minimized costs.
- Short Release Cycle Deployments: This would help to increase revenue by releasing new value-generating features fast.
- Detecting Vulnerabilities Early: will protect the company from egregious company situations, thus, protecting revenue and avoiding uncalled for expenses.
- Automated Tests: protects revenue by enforcing code quality and security, reducing downtime from a deploy-related crash or a massive single point of failure.
- Minimize Technical Debt: technical debt may not be eliminated fully, but a CI/CD pipeline will help reduce it thus, reducing cost and protecting revenue.
- Monitoring & Logging is a benefit as well in the sense that it will enable our team to respond properly to overall system failures and thus ensuring quality client satisfaction which means increased revenue from uninterrupted usage of the end products.
- Automated Resource Clean-up: would leverage on preservation of cloud compute clusters hence reducing cost, avoiding cost, and improving revenue.