DevOps é uma filosofia cujo objetivo central é eliminar a competição entre as equipes de desenvolvimento e as de operações e infraestrutura. Se você não está familiarizado com o setor de Tecnologia da Informação (TI), é importante entender que, por muitos anos, as organizações mantiveram distintas as áreas de Desenvolvimento de Software e as de Operação e Infraestrutura de Sistemas. A equipe de Desenvolvimento cria produtos que são implantados em servidores, computadores e na nuvem, os quais são suportados pelas equipes de Operação.

Lamentavelmente, por terem sempre funcionado como entidades distintas, surgiu uma espécie de animosidade entre elas. A equipe de Desenvolvimento critica a equipe de Operações por ser morosa, excessivamente cautelosa e obstruir o progresso dos programas. Por outro lado, a equipe de Operações vê a equipe de Desenvolvimento como despreparada, sem compreensão do ambiente e ineficaz. Essa situação prevaleceu durante anos até que perceberam o que agora parece claro. Uma área depende completamente da outra e não conseguem operar de forma independente. Foi nesse cenário que nasceu a ideia de DevOps.

A meta principal é unir os grupos responsáveis pelo desenvolvimento e pela operação de software, com o intuito de atender rapidamente e com qualidade às demandas dos clientes de tecnologia da informação. Devops compreende a utilização de ideias, métodos e ferramentas que aprimoram a habilidade de uma organização de oferecer produtos e serviços de maneira eficaz e com eficiência. Ademais, isso influencia a interação entre as equipes de Desenvolvimento e Operações, promovendo uma cultura colaborativa entre os especialistas dessas áreas.

Existem diversas vantagens ao implementar a cultura DevOps. Algumas delas incluem:

Harmonia entre setores que precisam trabalhar em conjunto

Integração entre profissionais, equipes, grupos e departamentos

Aprimoramento da qualidade dos serviços e produtos disponibilizados

Redução do intervalo entre a concepção da ideia ou pedido do cliente e a finalização da entrega

Aumento na confiabilidade, segurança e capacidade de expandir tanto do produto final quanto da infraestrutura que o sustenta

Troca de conhecimento e aprimoramento contínuo de todos os participantes

Desvantagens do DevOps

Nem tudo são vantagens. Existem alguns aspectos que merecem atenção ao se considerar o DevOps:

A complexidade tende a crescer, já que gerenciar setores isolados é bem mais simples do que fazê-lo de forma integrada.

Enfrentar a situação de gestores que não terão "total controle" sobre suas equipes.

As limitações das pessoas, visto que o mundo da tecnologia da informação é vasto. Esperar que indivíduos sejam especialistas em Cobol, Java, Python, Kubernetes, Banco de Dados, Jenkins, Docker, Nuvem e Inteligência Artificial é um tanto irreal.

O custo da automação pode ser elevado, pois, além dos investimentos em ferramentas, é necessário considerar o custo do aprendizado para utilizá-las.

CI/CD: Conceito e Aplicações

Visão Geral do CI/CD

CI/CD, que significa Integração Contínua (CI) e Entrega Continua/Implantação Contínua (CD), são métodos fundamentais no desenvolvimento de software contemporâneo. Estas abordagens têm como objetivo aprimorar a qualidade do código, otimizar o tempo de desenvolvimento e assegurar que as alterações no código sejam implementadas de forma ágil e contínua. A implementação de CI/CD tornou-se uma parte vital do ciclo de vida do software, especialmente em contextos ágeis, automatização e DevOps.

Integração Contínua (CI)

Integração Contínua é uma abordagem que envolve a junção frequente (tipicamente múltiplas vezes ao dia) do código criado por diferentes desenvolvedores em um repositório comum. O objetivo principal desta prática é identificar e resolver rapidamente problemas de integração, o que pode incluir conflitos de código, falhas em dependências ou erros introduzidos ao longo do desenvolvimento.

Os elementos principais da CI são:

Repositório de código unificado: Os desenvolvedores realizam o "commit" de suas alterações em um repositório centralizado.

Automatização de compilações e testes: Sempre que há um commit no repositório, o processo de compilação do código e a execução de testes automáticos são acionados. Isso assegura que, se um novo código comprometer a funcionalidade existente, o problema será detectado rapidamente.

Retorno imediato: Os desenvolvedores recebem retorno instantâneo sobre o status do código, permitindo a correção rápida de falhas.

Ferramentas populares de CI incluem Jenkins, CircleCI, GitLab CI e Travis CI.

Entrega Contínua (CD) e Implantação Contínua (CD)

Apesar de frequentemente serem usados como sinônimos, existem diferenças sutis entre Entrega Contínua e Implantação Contínua:

Entrega Contínua: Refere-se à prática de manter o código em um estado que possa ser liberado para produção a qualquer momento. Isso significa que, após a realização bem-sucedida de testes automáticos e verificação da qualidade do código, o código está pronto para implantação, porém ainda requer ação manual para a sua efetiva implementação.

Implantação Contínua: Esta é uma expansão da entrega contínua, onde a implantação em produção é inteiramente automatizada. Assim, toda vez que uma modificação é integrada ao repositório e aprovada nos testes, ela é implantada automaticamente em produção, sem necessidade de intervenção manual.

Essas abordagens auxiliam as equipes na rápida e confiante liberação de novas funcionalidades, garantindo que quaisquer erros sejam rapidamente detectados e corrigidos.

Ferramentas comuns de CD incluem Spinnaker, Jenkins, GitLab CI/CD, Bamboo, entre outras.

Ferramentas Utilizadas no CI/CD

Besides the practices of CI/CD, the selection of suitable tools is crucial for a successful deployment. Let’s discuss some of the most widely used tools that enhance these practices.

Docker

Docker is a containerization platform that allows applications and their execution environments to be packaged and shared. With Docker, a developer can create a container that includes the application code, dependencies, and necessary configurations to run the application consistently, regardless of the environment—whether it's development, testing, or production.

In the realm of CI/CD, Docker enables development and operations teams to work with the same configuration across all steps of the pipeline. This ensures that the application operates in exactly the same way in various environments, resolving the age-old issue of "it worked on my machine." Additionally, Docker simplifies the creation of isolated test environments where new code versions can be validated without impacting the production environment.

Git

Git is a widely used version control system designed to track modifications in source code. It allows multiple contributors to collaborate effectively by creating branches to introduce new features, and then merging those branches back into the main one.

Within the CI/CD framework, Git constitutes the backbone of continuous integration. Git repositories, such as GitHub, GitLab, and Bitbucket, serve as the storage and versioning locations for code. These repositories also integrate seamlessly with CI/CD tools like Jenkins and CircleCI, which trigger the build and testing processes whenever a change is made to the repository. Thus, Git and CI/CD work in tandem to ensure that changes are rapidly tested and incorporated.

Jenkins

Jenkins is an automation tool that streamlines the execution of CI/CD pipelines. It is widely employed to integrate and automate various phases of the software lifecycle, which includes building, testing, deploying, and monitoring.

Jenkins is customizable through plugins, allowing it to connect with a variety of other tools and services, such as Docker, Git, Kubernetes, and many more. It can be set up to monitor a Git repository, build the code whenever a new commit occurs, run automated tests, and even deploy the application to a production environment.

Embora não se classifique como uma verdadeira ferramenta de CI/CD, o Kubernetes tornou-se imprescindível em cenários de implementação contínua, especialmente no contexto de contêineres. Kubernetes é um sistema de gerenciamento de contêineres que facilita a instalação, o dimensionamento e a supervisão de aplicativos empacotados em contêineres. Ele proporciona aos desenvolvedores e às equipes de operações uma forma mais eficaz de gerenciar a infraestrutura, permitindo o escalonamento automático dos aplicativos conforme necessário e garantindo disponibilidade constante.

Vantagens do CI/CD

Implementar CI/CD traz diversos benefícios para as equipes de desenvolvimento, tais como:

Identificação rápida de falhas: Através da aplicação de testes automatizados e da integração regular do código, falhas podem ser descobertas precocemente no ciclo de desenvolvimento, economizando tempo e esforço na resolução.

Manutenção da qualidade do código: Ao adotar testes automáticos e avaliações de qualidade durante todas as fases do processo de desenvolvimento, as equipes asseguram que o código entregue mantenha um alto padrão.

Maior automação e produtividade: CI/CD diminui a necessidade de intervenções manuais e ajuda a reduzir erros cometidos por pessoas. Ademais, permite que as equipes se dediquem mais à criação de novas funcionalidades e menos a tarefas de construção e implantação repetitivas.

Implementação ágil de novas funcionalidades: Com a automação nos procedimentos de implantação, as organizações conseguem lançar novas funcionalidades, correções e atualizações de maneira muito mais rápida, proporcionando um valor mais imediato aos usuários.

Minimização de riscos: A implementação regular de alterações pequenas diminui a probabilidade de grandes falhas, pois os problemas podem ser detectados e solucionados de forma ágil.

A combinação de práticas como CI/CD e ferramentas como Docker, Git e Jenkins tem transformado a forma como o software é desenvolvido e entregue. Essas abordagens promovem maior eficiência, qualidade e colaboração entre equipes de desenvolvimento e operações, resultando em ciclos de desenvolvimento mais rápidos e aplicações mais confiáveis. Ao adotar essas ferramentas e práticas, as equipes podem garantir que entreguem valor de maneira contínua e sem interrupções, atendendo às necessidades do mercado de forma ágil e eficiente.