Edson Mitsuru Yamashita Yonamine RA: 0030482221041

Introdução

O DevOps é basicamente uma combinação de duas palavras: Desenvolvimento e Operações. DevOps é uma cultura que implementa a tecnologia para promover a colaboração entre a equipe de desenvolvedores e a equipe de operações para implantar o código na produção com mais rapidez, de forma automatizada e repetível. Envolve desenvolvimento de software iterativo, automação e implantação de infraestrutura programável. O DevOps enfatiza confiança, melhor comunicação e alinhamento de projetos tecnológicos com requisitos de negócios. O ciclo de vida do DevOps geralmente inclui etapas como planejamento, codificação, construção, teste, lançamento, implantação, operação, monitoramento e retorno ao planejamento.

DevOps – Conceitos, práticas, ferramentas, benefícios e exemplos

Conceitos e práticas

O DevOps enfatiza a construção de confiança e um melhor relacionamento entre desenvolvedores e administradores de sistema. Isso ajuda a organização a alinhar os projetos tecnológicos aos requisitos do negócio. As mudanças implementadas geralmente são pequenas e reversíveis, o que toda a equipe começa a compreender. Ele é visualizado como um loop infinito que compreende as etapas: planejar, codificar, construir, testar, liberar, implantar, operar, monitorar, depois voltar ao planejamento e assim por diante, é chamado de o ciclo de vida DevOps, abaixo estão as definições de cada etapa:

Planejar: Determinar as necessidades comerciais e coletar as opiniões do usuário final por profissionais deste nível do ciclo de vida DevOps.

Código: Neste nível é desenvolvido o código do mesmo e para simplificar o design, a equipe de desenvolvedores utiliza ferramentas e extensões que cuidam dos problemas de segurança.

Construir: Após a parte de codificação, os programadores utilizam diversas ferramentas para o envio do código ao código-fonte comum.

Teste: Este nível é muito importante para garantir a integridade do software. Vários tipos de testes são realizados, como testes de aceitabilidade do usuário, testes de segurança, testes de velocidade e muito mais.

Release: Neste nível tudo está pronto para ser implantado no ambiente operacional.

Implantar: neste nível, a infraestrutura como código auxilia na criação da infraestrutura operacional e posteriormente publica a construção usando várias ferramentas de ciclo de vida DevOps.

Operar: Neste nível, a versão disponível está pronta para uso dos usuários. Aqui, o departamento cuida da configuração e implantação do servidor.

Monitorar: A observação é feita neste nível que depende dos dados coletados do comportamento do consumidor, da eficiência das aplicações e de diversas outras fontes.

Melhores práticas a seguir:

* Implementar painel automatizado
* Mantenha toda a equipe unida
* Permita que o DevOps seja uma mudança cultural
* Seja paciente com os desenvolvedores
* Mantenha uma unidade centralizada
* Construa uma infraestrutura flexível

Ferramentas para DevOps

Para que tenha um bom desenvolvimento na produção utilizando devOps, abaixo estão alguns exemplos de ferramentas para seu uso:

**Jenkins** - Jenkins é um servidor de automação de código aberto e gratuito que ajuda a automatizar processos de desenvolvimento de software, tais como construção, facilitação de CI/CD, implantação e testes. Esta ferramenta DevOps torna fácil para as equipes monitorar tarefas repetidas, integrar facilmente as mudanças e identificar problemas rapidamente.

**Docker** - Líder em contentores de software, o Docker é utilizado por mais de 11 milhões de desenvolvedores em todo o mundo. A Solomon Hykes é sua autora original e foi lançada em 2013 pela Docker, Inc., uma das maiores empresas de desenvolvimento de software do mundo. Como uma ferramenta DevOps, Docker ajuda os desenvolvedores a construir, empacotar e depois implantar os códigos com facilidade e rapidez através de recipientes com as dependências necessárias em vez de máquinas virtuais. Ele elimina atividades de configuração mundanas e fomenta a colaboração eficaz da equipe.

**Puppet** - Desenvolvido pela Puppet, Inc. e fundado em 2005 por Luke Kanies, Puppet é uma ferramenta de código aberto para gerenciamento de configuração de software. Esta ferramenta de automação DevOps ajuda a gerenciar diferentes estágios do ciclo de vida do software, como o provisionamento da infraestrutura de TI, correção, configuração e gerenciamento de componentes de software e SO em infraestruturas de nuvem e centros de dados.

**Apache Maven** - Desenvolvida pela Apache Software Foundation e lançada em 2004, a Maven é uma ferramenta de automação de construção eficiente. Escrito em Java, é utilizado principalmente para projetos baseados em Java e atua como uma ferramenta de gerenciamento e compreensão de projetos. Como uma ferramenta DevOps, ela ajuda a gerenciar a construção, documentação e relatórios de um projeto. Além dos projetos baseados em Java, você também pode usar a Maven para desenvolver e gerenciar projetos de software escritos em Ruby, C#, Scala, e outros.

**Gradle** - Acelere sua produtividade de desenvolvimento de software usando Gradle. Isto também está entre as ferramentas DevOps de código aberto para automação de construção, especialmente para aplicativos multilíngües ou desenvolvimento de software. Gradle é escrito em Java, Kotlin e Groovy e foi lançado em 2007. É usada para automatizar o desenvolvimento, teste e implantação de software em velocidades melhores.

**CircleCI** - Não importa o tamanho de sua equipe DevOps, esta ferramenta nativa da CI/CD CircleCI oferece confiabilidade e velocidade para seu processo de desenvolvimento de software. Você pode desenvolver e implantar rapidamente software de alto desempenho em sua infraestrutura ou na nuvem em escala. O CircleCI foi fundado em 2011, e agora funciona em torno de 35M+ construções por mês. É usado por empresas líderes como Spotify, PagerDuty, Ford Motor Company, Samsung, e mais.

**Bamboo** - A equipe DevOps escolheu a Bamboo by Atlassian para integração contínua, implantação contínua e entrega de aplicativos de software. Você pode unir seus builds, releases e testes automatizados como um único fluxo de trabalho. Como ferramenta DevOps, ela ajuda você a criar planos de construção em vários estágios, estabelecer gatilhos, e alocar agentes para construções críticas, bem como implantações.

**TeamCity** - Lançado em 2006, o TeamCity by JetBrains é uma ferramenta robusta de integração contínua para as equipes DevOps. É uma ferramenta de uso geral CI/CD DevOps que oferece maior flexibilidade para diferentes tipos de desenvolvimento e práticas de fluxo de trabalho. Desde desenvolvedores e engenheiros DevOps até gerentes e administradores; TeamCity é amada por todos. Você pode construir, implantar e testar aplicativos, recipientes e pacotes de diferentes tipos, sejam eles multi-nuvem, multi-linguagem ou multiplataforma.

**Travis CI** - Mais adequado para projetos open source, o Travis CI é uma ferramenta eficiente de integração contínua (CI) para desenvolver e testar aplicativos hospedadas no GitHub ou Bitbucket. Foi escrito em Ruby e tem uma licença MIT. Este serviço hospedado na nuvem detecta todos os novos códigos cometidos automaticamente e quantos códigos são movidos para o seu repositório GitHub para ajudá-lo a construir o software e testá-lo de acordo.

**Buddy** - Se você encontrar uma ferramenta DevOps eficaz para que a CI/CD construa seu software mais rapidamente, Buddy pode ser seu melhor ‘amigo’. Esta ferramenta ajuda na integração e implantação contínua junto com o feedback. É adequado para projetos que utilizam códigos dos repositórios GitHub e Bitbucket. Também permite utilizar containers Docker além de estruturas e linguagens pré-instaladas para construir o software, e também monitora e envia alertas.

**Git** - Uma das ferramentas DevOps mais reconhecidas, Git é um software de controle de versão gratuita e de código aberto. Lançado em 2005, foi escrito em C, Perl, Shell, e Tcl, e originalmente escrito por Linus Torvalds. Esta ferramenta de gerenciamento de código fonte distribuído (SCM) é utilizada para rastrear mudanças em arquivos para desenvolvimento de software e coordenar eficazmente o trabalho entre programadores. Ela visa aumentar a velocidade, suporte e integridade dos dados para fluxos de trabalho não lineares e distribuídos.

**GitHub** - Milhões de empresas e desenvolvedores confiam no GitHub para construir, gerenciar e enviar seus softwares. Esta é a plataforma de desenvolvimento mais avançada e maior do mundo. Ela fornece hospedagem web para seu desenvolvimento de software, assim como seu controle de versão utilizando Git. Escrito em Ruby, C, Go, e ECMAScript, foi lançado em 2008. Atualmente, é utilizado por mais de 56 milhões de usuários e mais de 3M+ organizações no mundo.

**Chef** - Uma das mais antigas ferramentas DevOps, o Chef é uma solução de gerenciamento de configuração de código aberto lançada em 2009. Você pode usar o Chef para agilizar a configuração e manutenção de seus servidores. Ele também pode se integrar com várias soluções baseadas em nuvem, tais como Amazon EC2, Microsoft Azure, Google Cloud Platform, etc. para configurar e fornecer novas máquinas automaticamente.

**Kubernetes** - Também referido como K8s, Kubernetes é um sistema de orquestração de contêineres de código aberto de produção para automatizar a implantação, gerenciamento e dimensionamento de software baseado em contêineres. Foi escrito em Go e desenvolvido originalmente pelo Google, e lançado em 2014, mas agora a Cloud Native Computing Foundation se encarrega de sua manutenção.

**Ansible** - Ansible é uma ferramenta de código aberto líder no fornecimento, implantação e gerenciamento de configuração de software que permite a infraestrutura como código. Escrito em Python, Shell, Ruby, e PowerShell, seu autor original é Michel DeHaan e foi lançado em 2012. Agora, a Red Hat adquiriu a Ansible.

Vantagens e Desvantagens de se usar DevOps

A utilização de DevOps pode trazer muitas vantagens para o projeto, algumas delas são:

1. **Entrega mais rápida:** o DevOps permite que as organizações lancem novos produtos e atualizações com mais rapidez e frequência, o que pode levar a uma vantagem competitiva.
2. **Colaboração aprimorada**: o DevOps promove a colaboração entre as equipes de desenvolvimento e operações, resultando em melhor comunicação, maior eficiência e redução de atrito.
3. **Qualidade aprimorada:** DevOps enfatiza testes automatizados e integração contínua, o que ajuda a detectar bugs no início do processo de desenvolvimento e a melhorar a qualidade geral do software.
4. **Maior automação:** o DevOps permite que as organizações automatizem muitos processos manuais, liberando tempo para trabalhos mais estratégicos e reduzindo o risco de erro humano.
5. **Melhor escalabilidade:** o DevOps permite que as organizações escalem sua infraestrutura de forma rápida e eficiente para atender às demandas em constante mudança, melhorando a capacidade de responder às necessidades de negócios.
6. **Maior satisfação do cliente:** o DevOps ajuda as organizações a fornecer novos recursos e atualizações mais rapidamente, o que pode resultar em maior satisfação e fidelidade do cliente.
7. **Segurança aprimorada:** o DevOps promove práticas recomendadas de segurança, como testes e monitoramento contínuos, que podem ajudar a reduzir o risco de violações de segurança e melhorar a segurança geral dos sistemas de uma organização.
8. **Melhor utilização de recursos:** o DevOps permite que as organizações otimizem o uso de recursos, incluindo hardware, software e pessoal, o que pode resultar em economia de custos e maior eficiência.

Mesmo tendo diversas vantagens, também temos o lado desvantajoso, sendo elas:

1. **Alto investimento inicial:** a implementação do DevOps pode ser um processo complexo e caro, exigindo investimentos significativos em tecnologia, infraestrutura e pessoal.
2. **Escassez de competências:** Encontrar profissionais DevOps qualificados pode ser um desafio e as organizações podem precisar de investir em programas de formação e desenvolvimento para desenvolver as competências necessárias nas suas equipas.
3. **Resistência à Mudança:** Alguns funcionários podem resistir às mudanças culturais e organizacionais necessárias para uma adoção bem-sucedida do DevOps, o que pode resultar em resistência, resistência à colaboração e redução da eficiência.
4. **Falta de padronização:** DevOps ainda é um campo relativamente novo e falta padronização em termos de metodologias, ferramentas e processos. Isso pode dificultar que as organizações determinem a melhor abordagem para suas necessidades específicas.
5. **Maior complexidade:** DevOps pode aumentar a complexidade da entrega de software, exigindo que as organizações gerenciem um maior número de peças móveis e integrem vários sistemas e ferramentas.
6. **Dependência de tecnologia**: o DevOps depende fortemente da tecnologia e as organizações podem precisar investir em uma variedade de ferramentas e plataformas para apoiar o processo de DevOps.
7. **Necessidade de melhoria contínua:** DevOps requer melhoria e adaptação contínuas, à medida que surgem novas tecnologias e melhores práticas. As organizações devem estar preparadas para adaptar e evoluir continuamente as suas práticas de DevOps para permanecerem competitivas.

Conclusão

Em resumo, DevOps representa uma abordagem cultural, metodológica e prática que visa melhorar a colaboração entre equipes de desenvolvimento de software (Dev) e operações de TI (Ops), com o objetivo final de acelerar a entrega de software de alta qualidade e confiabilidade. Ao integrar desenvolvimento, teste, implantação e operações em um ciclo de vida de desenvolvimento contínuo, DevOps promove uma cultura de colaboração, automação, feedback contínuo e responsabilidade compartilhada. Além de diversos benefícios do DevOps, incluindo ciclos de desenvolvimento mais curtos, implantações mais rápidas e confiáveis, maior qualidade de software, maior flexibilidade e capacidade de resposta às mudanças do mercado.

No entanto, a implementação bem-sucedida do DevOps requer mais do que apenas ferramentas e automação; requer uma mudança cultural significativa, incluindo colaboração, transparência, confiança e compartilhamento de conhecimento entre equipes. Além disso, a adoção de práticas ágeis, como integração contínua, entrega contínua e infraestrutura como código, é fundamental para o sucesso do DevOps.

Referências

Site GeeksforGeeks, 2023, Link: <https://www.geeksforgeeks.org/introduction-to-devops/>

Site Kinsta, Descubra as 30 Principais Ferramentas DevOps para Observar em 2024, link: <https://kinsta.com/pt/blog/ferramentas-devops/>