cesae digital Centro para o Desenvolvimento de Competências Digitais

Curso DevOps: Automação e Integração Contínua

Introdução à DevOps

Formador: Patrício dos Santos
E-mail: *hello@psantos.dev*
LinkedIn: *https://linkedin.com/in/psantos11*
X (ex-Twitter): *https://x.com/psantos_ao*

INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
AEP ASSOCIAÇÃO EMPRESARIAL DE PORTUGAL Câmara de Comércio e Indústria
PORTUGAL 2030
Cofinanciado pela União Europeia

Objetivos

- Compreender o que é DevOps e sua importância no desenvolvimento de software.
- Explorar os princípios, práticas e pilares fundamentais de DevOps.
- Identificar os benefícios e desafios da adoção de DevOps.
- Conhecer as ferramentas e fluxos de trabalho comuns em DevOps.

1. O que é DevOps?

1.1 Definição

DevOps é uma cultura, conjunto de práticas e filosofia que promove a colaboração entre equipas de Desenvolvimento (Dev) e Operações (Ops) para acelerar a entrega de software de alta qualidade. O termo "DevOps" combina "Development" e "Operations" e busca eliminar silos, automatizar processos e melhorar a eficiência no ciclo de vida do desenvolvimento de software.

"*DevOps é sobre pessoas, processos e ferramentas trabalhando juntos para entregar valor ao cliente continuamente.*" – Gene Kim, autor de The Phoenix Project.

Tradicionalmente, o desenvolvimento de software caracterizava-se por uma separação rígida entre as equipas de desenvolvimento (Dev) e as equipas de operações (Ops). As equipas de desenvolvimento focavam-se na criação de novas funcionalidades, enquanto as equipas de operações concentravam-se na estabilidade e na fiabilidade dos sistemas. Esta divisão resultava frequentemente em:

- Longos ciclos de desenvolvimento e implementação
- Múltiplas falhas em ambiente de produção
- Comunicação deficiente entre departamentos
- Atribuição mútua de culpas quando surgiam problemas
- Resistência à mudança por parte das equipas de operações
- Desconhecimento dos requisitos operacionais por parte dos programadores

1.2 Origem de DevOps

O DevOps emergiu no final dos anos 2000, impulsionado por movimentos como o Agile e pela crescente necessidade de entregas de software mais rápidas e fiáveis. Esta abordagem revolucionária surgiu como resposta às frustrações que assolavam a indústria de tecnologia.

O termo "DevOps" foi cunhado por Patrick Debois e Andrew Shafer, combinando as palavras "desenvolvimento" e "operações". Inicialmente, o conceito espalhou-se através de comunidades online, blogues e conferências.

Empresas pioneiras como Flickr, Netflix e Etsy começaram a adotar práticas de DevOps logo no início, demonstrando resultados impressionantes em termos de velocidade de entrega e qualidade de software. A Flickr, por exemplo, passou de implementações bimensais para implementações diárias, tornando-se um caso de estudo frequentemente citado.

À medida que o conceito amadurecia, o DevOps deixou de ser apenas uma filosofia para se tornar um conjunto de práticas concretas, apoiadas por ferramentas específicas de automatização, integração contínua e monitorização que facilitavam a colaboração eficaz entre as equipas de desenvolvimento e operações.

Em Portugal, a adoção do DevOps iniciou-se mais tardiamente, por volta de 2012-2013, principalmente através de multinacionais com presença no país. Empresas como a Critical Software e a Farfetch estiveram entre as primeiras organizações portuguesas a implementar práticas de DevOps com sucesso, contribuindo para a disseminação desta abordagem no mercado nacional.

1.3 Por que DevOps?

Problemas tradicionais:

- Conflitos entre Dev (foco em mudanças rápidas) e Ops (foco em estabilidade).
- Entregas lentas devido a processos manuais.
- Falhas frequentes em produção por falta de integração.

Soluções com DevOps:

- Integração contínua e entrega contínua (CI/CD).
- Automação de processos repetitivos.
- Cultura de colaboração e responsabilidade compartilhada.

2. Princípios Fundamentais de DevOps

DevOps é sustentado por princípios que orientam sua adoção. Os principais são baseados no modelo CALMS:

2.1 CALMS Framework

Cultura: Promover colaboração, confiança e responsabilidade partilhada entre equipas.

Automação: Automatizar processos como testes, builds, deploys e monitoramento para reduzir erros e acelerar entregas.

Lean: Aplicar princípios enxutos para eliminar desperdícios, otimizar fluxos e focar no valor ao cliente.

Measrement (Mensuração): Monitorar métricas (ex.: tempo de deploy, taxa de falhas) para tomar decisões baseadas em dados.

Sharing (Partilha): Partilhar conhecimento, ferramentas e práticas entre equipes para aprendizado contínuo.

2.2 Os Três Caminhos (Three Ways)

Conceito apresentado no livro The DevOps Handbook:

- **Fluxo (Flow):** Otimizar o fluxo de trabalho do desenvolvimento até a produção, reduzindo gargalos.
- **Feedback**: Criar loops de feedback rápidos para identificar e corrigir problemas rapidamente.
- **Aprendizado Contínuo**: Fomentar experimentação, aprendizado com falhas e melhoria contínua.

3. Pilares de DevOps

Os pilares abaixo são práticas essenciais para implementar DevOps:

3.1 Integração Contínua (CI)

Desenvolvedores integram código frequentemente em um repositório partilhado, com testes automatizados para validar mudanças.

Esta prática ajuda a detetar erros rapidamente, melhora a qualidade do código.

Ferramentas: Jenkins, GitHub Actions, GitLab CI, CircleCI.

3.2 Entrega Contínua (CD)

Extensão do CI, onde o código validado é automaticamente preparado para *deploy* em produção, mas requer aprovação manual.

Ajuda a reduzir riscos em deploys, e permite entregas frequentes.

Ferramentas: ArgoCD, Spinnaker.

3.3 Implantação Contínua

Código validado é automaticamente implantado em produção sem intervenção manual.

Isso maximiza a velocidade de entrega, ideal para empresas com alta maturidade em automação.

Exemplo: Netflix usa implantação contínua para lançar atualizações várias vezes ao dia.

3.4 Infraestrutura como Código (IaC)

Gestão de infraestrutura (servidores, redes) usando código, permitindo provisionamento e configuração automatizados.

Com esta prática, ganha-se consistência, escalabilidade, recuperação rápida de desastres.

Ferramentas: Terraform, Ansible, AWS CloudFormation.

3.5 Monitoramento e Observabilidade

Coletar, analisar e visualizar dados para entender o desempenho de sistemas e identificar problemas. Permite a resolução proativa de falhas, melhoria da experiência do utilizador.

Ferramentas: Prometheus, Grafana, ELK Stack.

4. Benefícios e Desafios de DevOps

4.1 Benefícios

Velocidade: Entregas mais rápidas com CI/CD.

Qualidade: Menos erros devido a testes automatizados.

Colaboração: Equipas mais alinhadas, reduzindo conflitos.

Escalabilidade: Infraestrutura gerida de forma eficiente.

Satisfação do cliente: Lançamentos de novas funcionalidades frequentes e confiáveis.

4.2 Desafios

Mudança cultural: Resistência à quebra de silos.

Complexidade técnica: Requer conhecimento em automação e de várias ferramentas.

Custo inicial: Investimento em treinamento e ferramentas.

Segurança: Integração de práticas de segurança (DevSecOps) pode ser desafiadora.

5. Ferramentas Comuns em DevOps

As ferramentas DevOps cobrem diferentes etapas do ciclo de vida do software. Abaixo, uma categorização com exemplos:

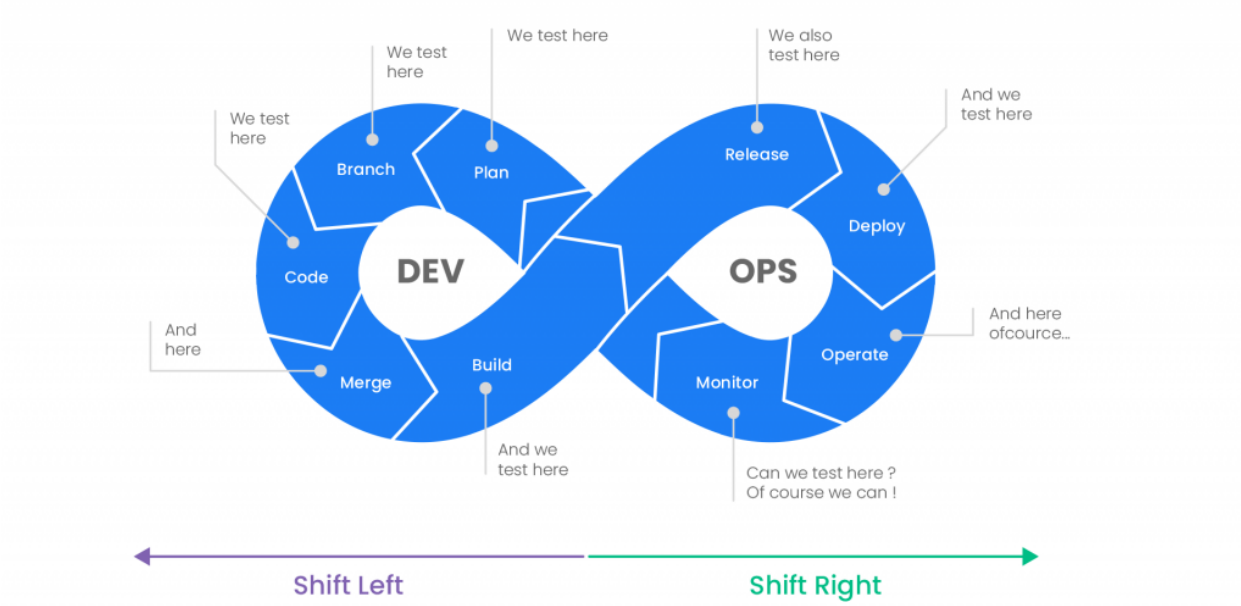
Categoria	Ferramentas
Controle de Versão	Git, GitHub, GitLab, Bitbucket
Integração Contínua	GitHub Actions, Jenkins, GitLab CI, CircleCI, Travis CI
Entrega/Implantação	ArgoCD, Spinnaker, AWS CodeDeploy
Infraestrutura como Código	Terraform, Ansible, Chef, Puppet
Contêineres/Orquestração	Docker, Kubernetes, OpenShift
Monitoramento	Prometheus, Grafana, New Relic, Datadog
Colaboração	Slack, Microsoft Teams, Jira

Nota: A escolha de ferramentas depende do contexto da organização (ex.: tamanho da equipe, orçamento, stack tecnológica).

6. Fluxo de Trabalho Típico em DevOps

Um fluxo de trabalho DevOps geralmente segue o Ciclo de Vida DevOps:

- **Planeamento**: Definir requisitos e planejar sprints (ex.: usando Jira).
- **Codificação**: Escrever código e armazená-lo em repositórios Git.
- **Build**: Compilar e integrar código com CI (ex.: Jenkins).
- **Testes**: Executar testes automatizados (unitários, integração, segurança).
- **Deploy**: Implantar em ambientes de staging/produção (ex.: Kubernetes).
- **Operação**: Monitorar sistemas em produção (ex.: Prometheus).
- **Feedback**: Analisar métricas e logs para melhorias contínuas.



7. Cultura DevOps: Pessoas e Processos

7.1 Mudança Cultural

- ✈ **Colaboração**: Dev e Ops trabalham como um time unificado.
- ✈ **Responsabilidade partilhada**: Todos são responsáveis pela qualidade e entrega.
- ✈ **Aprendizado contínuo**: Experimentar, falhar rápido e aprender com erros.

7.2 Papéis em DevOps

Engenheiro DevOps: Automatiza pipelines, faz a gestão da infraestrutura.

Engenheiro de Confiabilidade do Site (SRE): Foca em estabilidade e escalabilidade.

Desenvolvedor: Escreve código com práticas DevOps (ex.: testes automatizados).

Especialista em Segurança: Integra práticas de DevSecOps.

7.3 Práticas Recomendadas

Realizar blameless post-mortems (análise de falhas sem culpas).

Adotar pair programming e revisões de código.

Implementar game days para simular falhas e testar resiliência.

8. DevOps na Prática: Estudos de Caso

8.1 Netflix

Líder em streaming, com milhões de utilizadores a nível global.

Práticas DevOps:

- Implantação contínua com Spinnaker.
- Infraestrutura como código com AWS.
- Monitoramento avançado com ferramentas como Chaos Monkey.

Resultado: Deploys diários, alta disponibilidade, escalabilidade.

8.2 Etsy

Plataforma de e-commerce para produtos artesanais.

Práticas DevOps:

- Cultura de experimentação e feedback rápido.
- CI/CD com ferramentas internas.
- Monitoramento detalhado de métricas.

Resultado: Redução de falhas em produção, entregas frequentes.

9. Conclusão

DevOps é mais do que ferramentas ou processos; é uma transformação cultural que alinha pessoas, tecnologia e objetivos de negócios. Para profissionais, dominar DevOps significa entender seus princípios, adotar práticas como CI/CD e IaC, e cultivar uma mentalidade de colaboração e aprendizado contínuo. Esta sessão inicial fornece a base para explorar tópicos mais avançados nas próximas aulas, como configuração de pipelines, orquestração de *containers* e DevSecOps.

10. Recursos Adicionais

Livros:

- The Phoenix Project (Gene Kim) – Introdução narrativa a DevOps.
- The DevOps Handbook (Gene Kim, Jez Humble, et al.) – Guia prático.
- Accelerate (Nicole Forsgren, Jez Humble, Gene Kim) – Dados sobre desempenho DevOps.

Comunidades:

- DevOpsDays (eventos globais).
- Grupos no Slack (ex.: DevOps Chat).
- Posts no X sobre #DevOps