# Vídeo 1 – O que é o DevOps

## Parte 1 – Problema: O “vale” entre desenvolvimento e operações

Pontos-chave:

* - Dev quer velocidade; Ops quer estabilidade.
* - Exemplo SoftWeb: metas conflitantes entre entregas rápidas e alta disponibilidade.
* - Impacto: atrasos, retrabalho e queda de satisfação do cliente.
* - Sintomas: deploys raros e dolorosos, muitos handoffs e silos.

Diagrama: `diagramas/v1\_p1\_vale\_dev\_ops.puml`

### Texto de fala:

Imagine dois times puxando em direções opostas: um acelerando para lançar funcionalidades e outro freando para manter a estabilidade. Esse choque de objetivos gera filas, aprovações intermináveis e um medo generalizado de mudar o sistema. É o cenário clássico de deploys raros e tensos, com longas janelas de mudança e muitas horas extras.

O caso SoftWeb ilustra bem: centenas de devs e poucos sysadmins, metas agressivas de novas features contra a exigência de 99%+ de disponibilidade. O resultado é um vale entre Dev e Ops que compromete a competitividade. Vamos investigar por que esse problema ocorre e como podemos superá‑lo com uma abordagem moderna.

## Parte 2 – Discussão: Por que o conflito acontece?

Pontos-chave:

* - Métricas desalinhadas (velocidade vs. estabilidade).
* - Processos em silos e comunicação limitada.
* - Falta de integração desde o planejamento.
* - Cultura e incentivos que reforçam o antagonismo.

Diagrama: `diagramas/v1\_p2\_causas\_conflito.puml`

### Texto de fala:

Quando cada equipe é medida por indicadores opostos, surge a política do 'não mexe no que está funcionando'. Sem integração desde o início, decisões técnicas e de infraestrutura são tomadas isoladamente, aumentando riscos e gerando retrabalho nas fases finais.

Além disso, a comunicação tende a ser assíncrona e burocrática: tickets, repasses e handoffs. Essa distância cria desconfiança e incentiva controles manuais que atrasam ainda mais o fluxo. Precisamos mudar objetivos, fluxo e cultura para reverter esse quadro.

## Parte 3 – Solução: A proposta cultural do DevOps

Pontos-chave:

* - DevOps = cultura + práticas + ferramentas.
* - Equipes integradas e responsabilidade ponta a ponta.
* - Objetivo comum: entregar valor rápido e com segurança.
* - Autonomia com alinhamento (guardrails).

Diagrama: `diagramas/v1\_p3\_integracao\_cultura.puml`

### Texto de fala:

DevOps propõe quebrar silos e alinhar incentivos: o time é dono do produto do código à operação. A colaboração começa no planejamento, passa pelo desenvolvimento e segue até a monitoração, criando feedbacks curtos e decisões embasadas por dados.

Esse modelo combina autonomia com guardrails: políticas automatizadas, padrões de qualidade e revisões leves. Assim, ganhamos velocidade sem perder estabilidade—o famoso 'mover rápido com segurança'.

## Parte 4 – Benefícios práticos e como medir

Pontos-chave:

* - Velocidade, escalabilidade, colaboração, confiabilidade e segurança.
* - Métricas: lead time, frequência de deploy, MTTR, taxa de falhas.
* - Automação reduz variabilidade humana.
* - Infraestrutura como código e testes em todos os níveis.

Diagrama: `diagramas/v1\_p4\_beneficios\_metricas.puml`

### Texto de fala:

Ao automatizar build, teste, segurança e deploy, o fluxo acelera e a variância cai. Passamos a medir o que importa: tempo de ideação ao deploy, frequência de releases, tempo médio de recuperação e taxa de mudanças que causam incidentes.

Essas métricas guiam melhorias contínuas e mostram o impacto no negócio: time‑to‑market menor, qualidade superior e experiência do usuário mais estável.

## Parte 5 – Exemplos e próximos passos

Pontos-chave:

* - Antes: deploy mensal e janelas longas; Depois: deploy diário.
* - Feature flags, rollback rápido e blue‑green/canary.
* - Começar pequeno: um serviço, um pipeline, um padrão.
* - Escalar práticas e padrões para todo o portfólio.

Diagrama: `diagramas/v1\_p5\_evolucao\_exemplos.puml`

### Texto de fala:

Times que adotam DevOps reduzem ciclos de meses para dias ou horas. Com feature flags e estratégias como blue‑green e canary, liberam valor continuamente sem expor todos os usuários a riscos de uma só vez.

O caminho começa pequeno: um pipeline exemplar, padrões reutilizáveis e um serviço piloto. A partir daí, escalamos o que funcionou e evoluímos as práticas com dados de produção.