DevOps in the Real World

Product Teams · Platform Engineering · Business Continuity · AI

ESTGD · VIII Jornadas de Informática · 29/04/2026

m-tech1

DevOps in the Real World

Building, delivering and operating production systems in the AI era

Francisco Santos Meireles
francisco.meireles@m-tech1.com

m-tech1 · Swiss Pharma DevSecOps lead · German Automotive Operations lead

m-tech1

Production is where architecture becomes real.

DevOps is not about installing tools. It is about creating the capability to change systems safely and recover when they fail.

m-tech1

Experience behind this talk

Swiss Pharma

DevSecOps lead for scientific product teams in regulated R&D environments.

German Automotive

OpenShift platform operations supporting multiple product teams.

m-tech1

International technology consulting: cloud, software, security and AI-enabled systems.

m-tech1

Where m-tech1 fits

m-tech1 as the international technology line within a broader business ecosystem.

Medinfotec

- Healthcare and MedTech operations
- Dental and clinical technology
- Business technology and digital platforms

m-tech1

- Cloud & DevSecOps
- Software engineering
- AI-enabled product delivery
- Business continuity engineering

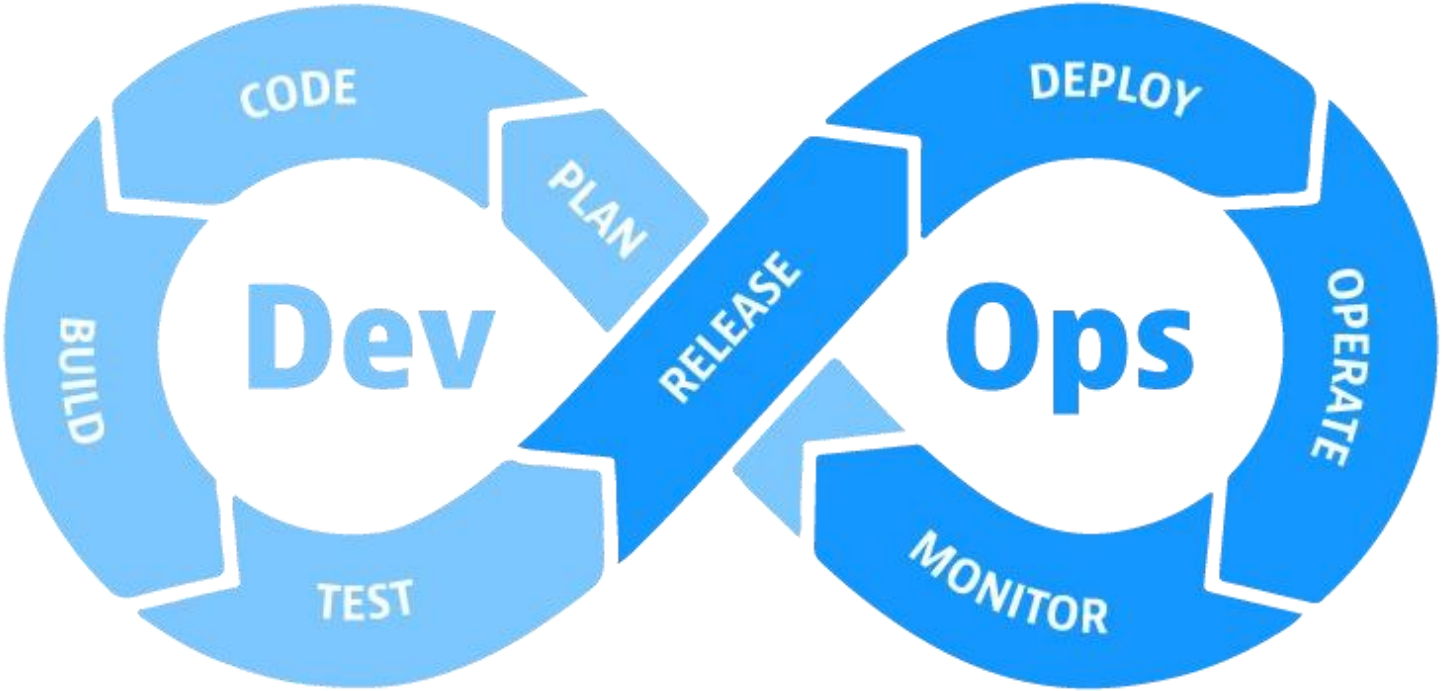
International direction

- Operating across borders
- Product mindset
- Engineering standards
- Industry-grade delivery

m-tech1

DevOps is the set of practices, tools and cultural philosophy to build, deliver, operate, and recover systems continuously, safely, and predictably in production.

m-tech1


CODE
PLAN
BUILD
Dev
TEST
RELEASE
DEPLOY
OPERATE
Ops
MONITOR

m-tech1

DevOps Core Principles

Flow & Speed

- Small, frequent changes
- Fast feedback loops

Ownership

- You build it, you run it
- Clear accountability

Automation

- Eliminate manual work
- CI/CD as baseline

Resilience

- Design for failure
- Recover fast, learn faster

DevOps is not about tools. It is about operating systems reliably in the real world.

m-tech1

Not one single flavour

It depends on what you own

Product Development

- CI/CD
- Testing
- Shipping features

speed + quality

Product Operations

- Monitoring
- Incident response
- Reliability

stability + recovery

Platform Egineering

- Developer platforms
- Infrastructure
- Self-service for teams

scale + standardization

m-tech1

DevOps, in practice

A production capability, not a tooling checklist.

Build

Create systems with operability in mind.

Deliver

Ship changes safely and repeatedly.

Operate

Know what is happening in production.

Recover

Restore service when reality breaks plans.

If your team cannot operate and recover the system, it is not really doing DevOps.

m-tech1

Swiss Pharma (R&D platform, regulated) case: product teams at scale

Context

- Scientific platforms and data-intensive workflows
- Multiple product teams, domains and services
- Regulated environment with high security expectations

Operational pressure

- Local optimization by teams creates global fragmentation
- Observability, security and deployment standards drift over time
- Product teams need enablement, not just central governance

m-tech1

The classic gap

Product teams are measured by features.

Production is measured by reliability.

- Feature velocity creates pressure to move fast.
- Operational excellence requires discipline.
- Without ownership, incidents become everybody's problem — and nobody's responsibility.

m-tech1

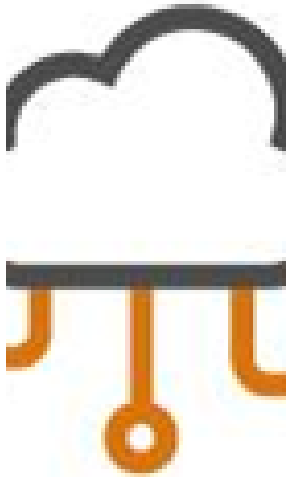
DevSecOps: security inside delivery

Traditional failure mode

- Security review happens too late
- Secrets and dependencies are discovered after release
- Patching becomes emergency work
- Compliance evidence is collected manually

Modern baseline

- Dependency and container scanning in CI/CD
- Secrets management and policy-as-code
- Signed artifacts and controlled registries
- Auditability built into the delivery process



m-tech1

DevSecOps as product enablement

Standardize

Common baseline for pipelines, runtime and controls.

Automate

Remove manual gates that create delay and risk.

Observe

Logs, metrics, traces and SLOs from day one.

Secure

Shift security into code, pipelines and runtime.

Recover

Runbooks, ownership and tested continuity.

m-tech1

German Automotive case: OpenShift operations

The problem was not “how to run Kubernetes”. The problem was how to make Kubernetes usable, stable and safe for product teams.

- Central platform ownership
- Cluster lifecycle management
- Guardrails, standards and self-service
- Product teams focus on products, not cluster internals

m-tech1

Platform Engineering: the scalable DevOps model

Classic “everyone does everything”

- Repeated infrastructure work
- Inconsistent pipelines and security controls
- Teams become accidental Kubernetes operators
- High cognitive load and uneven reliability

Platform-as-a-product

- Golden paths and paved roads
- Self-service with guardrails
- Shared observability and deployment standards
- Product teams retain ownership of outcomes

If this product fails, what will stop?

m-tech1

Business Continuity is the ability to operate through failure

Detect
Monitoring, alerts and clear service health signals.

Triage
Incident roles, severity and communication path.

Degrade
Fallbacks, feature flags and graceful failure.

Recover
Restore service, data and customer journeys.

Learn
Postmortems, fixes and prevention work.

m-tech1

Observability is the control tower

Without it, debugging is guesswork.

Metrics

How the system behaves over time.

Logs

What happened, where and why.

Traces

How requests move across services.

SLOs translate technical signals into business expectations.

m-tech1

AI can write code
assist decisions
take actions

But responsibility still belongs to humans!

m-tech1

AI changes DevOps — but it does not remove responsibility

- AI can accelerate coding, testing, documentation and incident analysis.
- It also increases supply-chain, prompt, data and over-automation risks.
- The winning teams will combine AI speed with engineering controls.

The question is not “Can AI generate it?”

The question is “Can we safely operate it?”

m-tech1

Real failures I’ve seen in production

- Manual deployments → hours of downtime
- Infrastructure not designed for failure → no scalability under load
- Exposed secrets → security incidents
- No proper testing → production breaks after release
- Untested disaster recovery → systems cannot recover
- Cloud costs out of control → infinite loops, uncapped usage
- Unused resources → silent cost leakage

m-tech1

If I were starting today

Observe early

Add telemetry before scale makes problems invisible.

Automate delivery

Simple CI/CD beats manual heroics.

Codify infra

Infrastructure as code creates repeatability.

Secure basics

Secrets, dependencies and permissions first.

Practice recovery

Backups, runbooks and drills must work.

m-tech1

m-tech1.com

Francisco Santos Meireles
francisco.meireles@m-tech1.com
+49 221 7474 9000