



THE MYSTERY OF DEVOPS AND WHAT IT'S ALL ABOUT

Technical Practices and Tooling

If we take a typical IT value stream of DesignDevelop-Deploy-Operate-Support, most of the technical practices and the tooling that you'll come across in the DevOps community, seem to focus on continuous integration / continuous delivery (CI/CD), which starts towards the back end of Development, when the developer commits the code to the repository for shipment, covers all of Deployment, and ends at the front end of Operations when the software is successfully running in production.

There are also technical practices and tools that support the actual Development, for example the adoption of trunk-based development rather than having multiple branches. Equally, there are also technical practices and tooling for Operations, for instance creating application logging telemetry that helps production. Some technical practices support Design, for instance using the strangler application pattern to safely evolve the enterprise architecture. There are also technical practices that support part of Support, for instance using telemetry to guide problem solving, but these are in the minority.

Principles and Culture

But there is more to doing things differently than technical practices and tooling alone.

DevOps, that mysterious thing that everybody seems to be talking about, is hard to pin down. People say that DevOps has helped them achieve fast, frequent and reliable delivery of software while at the same time providing resilient operational IT services, and fostering a healthy IT workforce. They say that they're doing things differently, but which things? And with which mindset?

Download more FREE DevOps resources at **Orbus Software** today!

www.orbussoftware.com/resources



There are also principles that guide when to do which things. There is a set of DevOps principles that contribute to almost every area of activity. For example, enabling organizational learning and a safety culture, and institutionalizing the improvement of daily work. Many of these principles have cultural implications, in other words, “how we do things around here”. What people do at work is based on their organisation’s set of interconnected artefacts, symbols, rituals, language, professed values, beliefs, assumptions and unspoken rules. These are the component parts of culture.

Changing “how we do things around here” – in other words, culture – is notoriously difficult. It requires a concerted effort to change many of these component parts at the same time in order to escape the inertia of the ‘system’.

Belief System

One of the key parts of culture is the set of beliefs that are based on interpretations of the environment and unexamined assumptions – “we’ve always done it that way”. DevOps has its own set of beliefs, some of which are clearly derived from DevOps’ foundations in Lean, Agile etc.

A good example is how change to production systems is regarded. Change is bad for stability, and stability is bad for change. The fewer the changes, the lower the risk to stability. The more stable the system, the more difficult it is to change. This is often the traditional IT service management belief: “less change is good”. But there’s another perspective. Reducing the size of the changes, reduces the risk of disruption. Having smaller changes means that you have more of them, so change happens more frequently. The more often you change, the more proficient you become.

The organization improves its change capability. This improved change capability leads, in turn, to a lower risk of disruption. The mental model has suddenly been reversed from “less change is good” to “more change is good”. This new mindset drives cultural change: “that’s how we do things around here”.

Examples of the DevOps belief system are:

- *Systems are complex: their intrinsic unpredictability means that it is often futile to analyse for root causes to problems – behaviour is emergent*
- *Servers are cattle, not pets: don’t try to repair a faulty server, replace it (the concept of immutability)*
- *Test in production: complex systems can’t be adequately tested in a pre-production environment, so testing in production is inevitable*
- *Fail forward: also related to the emergent nature of complex systems, rollback is often a dangerous illusion*
- *Be mean to your systems: get the Chaos Monkey to break the systems so that you can build resilience (“if it hurts, do it more”)*
- *Trust the experts: allow the team to decide when to release – peer review, not a Change Advisory Board, leads to better performance*
- *Stop and swarm: when things go wrong, stop production immediately and tackle the problem from multiple perspectives*

Summary

DevOps’ technical practices and tooling drive Deployment, contribute significantly to parts of Development and Operations, and contribute to a lesser degree to Design and Support. More fundamentally, DevOps’ principles help change the underlying culture by introducing new mental models. As such DevOps can be regarded as a belief system.





Phone: +44 (0)203 824 2907 • **Email:** enquiries@orbussoftware.com • **Website:** www.orbussoftware.com