UTT

UNIVERSIDAD TECNOLÓGICA DE TIJUANA

GOBIERNO DE BAJA CALIFORNIA

TEMA:

Introduction a DevOps

BY:

Derian Omar Tarango Mendez

GROUP:

10 B

COURSE:

Software development process management

PROFESSOR:

Ray Brunett Parra Galaviz

Tijuana, Baja California, 07 of january 2025

What is DevOps?

DevOps Model Defined

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization’s ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.

How DevOps Works

Under a DevOps model, development and operations teams are no longer “siloed.” Sometimes, these two teams are merged into a single team where the engineers work across the entire application lifecycle, from development and test to deployment to operations, and develop a range of skills not limited to a single function

In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle. When security is the focus of everyone on a DevOps team, this is sometimes referred to as DevSecOps.

Benefits of DevOps

Speed

Move at high velocity so you can innovate for customers faster, adapt to changing markets better, and grow more efficient at driving business results. The DevOps model enables your developers and operations teams to achieve these results. For example, microservices and continuous delivery let teams take ownership of services and then release updates to them quicker.

Rapid Delivery

Increase the frequency and pace of releases so you can innovate and improve your product faster. The quicker you can release new features and fix bugs, the faster you can respond to your customers’ needs and build competitive advantage. Continuous integration and continuous delivery are practices that automate the software release process, from build to deploy.

Scale

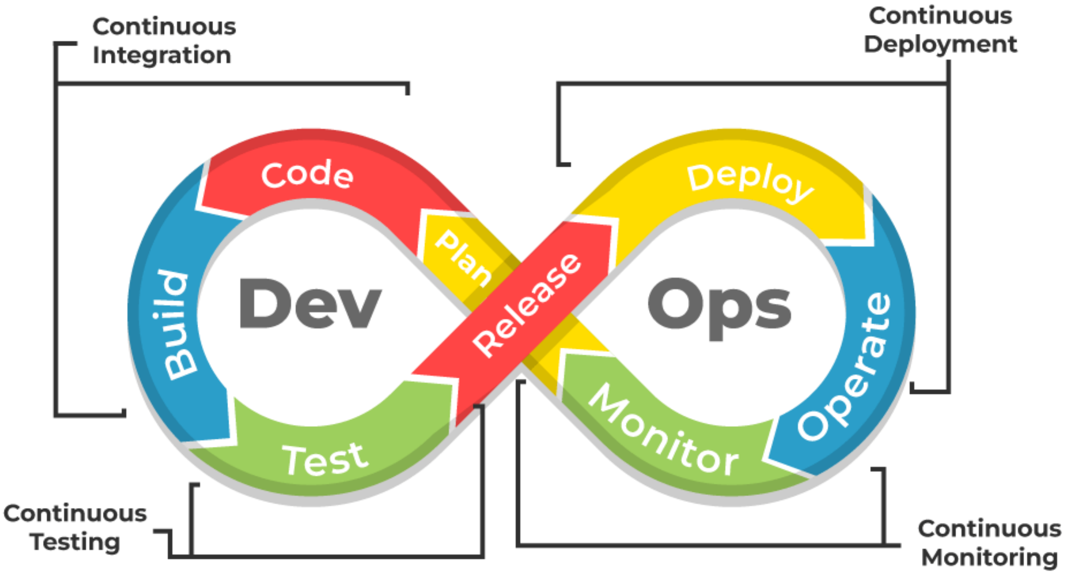
Operate and manage your infrastructure and development processes at scale. Automation and consistency help you manage complex or changing systems efficiently and with reduced risk. For example, infrastructure as code helps you manage your development, testing, and production environments in a repeatable and more efficient manner.

Improved Collaboration

Build more effective teams under a DevOps cultural model, which emphasizes values such as ownership and accountability. Developers and operations teams collaborate closely, share many responsibilities, and combine their workflows. This reduces inefficiencies and saves time (e.g. reduced handover periods between developers and operations, writing code that takes into account the environment in which it is run).

Why DevOps Matters

Software and the Internet have transformed the world and its industries, from shopping to entertainment to banking. Software no longer merely supports a business; rather it becomes an integral component of every part of a business. Companies interact with their customers through software delivered as online services or applications and on all sorts of devices. They also use software to increase operational efficiencies by transforming every part of the value chain, such as logistics, communications, and operations. In a similar way that physical goods companies transformed how they design, build, and deliver products using industrial automation throughout the 20th century, companies in today’s world must transform how they build and deliver software.



Core Principles of DevOps

Automation

- Automating repetitive tasks is central to DevOps. This includes processes like testing, integration, deployment, and infrastructure provisioning. Tools like Jenkins, Ansible, and Kubernetes are commonly used for automation.

Continuous Everything

- **Continuous Integration (CI):** Developers frequently merge code changes into a shared repository. Automated testing ensures the new code doesn’t break the existing system.
- **Continuous Delivery (CD):** Code is automatically prepared for deployment after passing CI.
- **Continuous Deployment:** Extends CD by deploying code automatically to production.

Collaboration

- DevOps emphasizes breaking down silos and encouraging seamless communication between development, operations, and other stakeholders. This is fostered by shared responsibility and cross-functional teams.

Feedback Loops

- Gathering and acting on feedback from monitoring, customer interactions, and team collaboration allows for iterative improvements.

DevOps Tools and Practices

1. **Version Control**
   - Tools: Git, GitHub, GitLab
   - Purpose: Track and manage code changes collaboratively.
2. **Configuration Management**
   - Tools: Puppet, Ansible, Chef
   - Purpose: Manage infrastructure as code (IaC), ensuring consistency across environments.
3. **Containerization**
   - Tools: Docker, Kubernetes
   - Purpose: Package applications and dependencies into containers for consistent deployment across environments.
4. **Monitoring and Logging**
   - Tools: Prometheus, Grafana, ELK Stack
   - Purpose: Real-time monitoring and troubleshooting.
5. **CI/CD Pipelines**
   - Tools: Jenkins, CircleCI, GitHub Actions
   - Purpose: Automate build, test, and deployment processes.