

# Implementing DevOps

## Duration

2 days

## Intended for

- Developers working in DevOps teams
- Operations staff supporting or working in DevOps teams

## Prerequisites

An understanding of fundamental Agile concepts and an understanding of DevOps concepts are necessary for you to get the maximum benefit from attending this course. These are covered in our Agile Fundamentals and Foundations of DevOps courses.

Experience using command lines is important as some of the hands-on work is performed in a Bash shell.

## Overview

Are you a developer or operations staff member working in a DevOps environment? Applying DevOps requires well-defined goals and a good understanding of the different tools available and which ones are right for you.

This course will provide you with hands-on experience of a DevOps environment: planning and setting the strategy; designing and implementing the pipeline; automating the provisioning and configuration of infrastructure and deployment of systems; monitoring the pipeline and the systems with telemetry; identifying bottlenecks; and applying continuous improvement to evolve the pipeline and the architecture.

## Learning Outcomes

By the end of this course you will:

- Be able to identify policies and processes to support DevOps and prepare a strategy for continuous delivery.
- Understand the impact of various architectural patterns on DevOps.
- Design a deployment pipeline and implement it.
- Automate the provisioning and configuring of environments in the cloud.
- Implement telemetry monitoring to support continuous improvement.
- Learn techniques for involving the customer in continuous improvement efforts.
- Know how to mature the pipeline, including evolving the architecture, using virtualisation and cloud computing, and ensuring compliance and governance is maintained.

## Content

Continuous improvement follows a Plan-Do-Study-Act cycle, and this course follows that cycle.

### Plan – identify the objectives.

- Identify what changes are needed in the culture, policies, and processes.
- Use Value Stream Mapping to identify waste in the process.
- Apply the Theory of Constraints to improve the process.
- Define an architecture, including the pipeline and its stages, to support the goals.

### Do – build the pipeline.

- Implement the pipeline (as code) in Go CD.
- Evaluate the benefits of automating each stage in the pipeline.
- Provision a VM in AWS EC2 with Terraform.
- Configure the VM and deploy the system with Ansible.

### Study – monitor the pipeline.

- Manage log data with the Elastic stack:
  - Collect log events with Logstash,
  - Aggregate them in Elasticsearch,
  - Analyse them with Kibana.
- Evaluate the benefits of different types of telemetry and the policies around it.
- Diagnose problems using the telemetry.
- Identify symptoms of security problems using the telemetry – DevSecOps.
- Evaluate policies and practices for support in production.
- Use formal techniques to elicit feedback from the users and customers.

# Implementing DevOps

## **Act – mature the pipeline**

- Evolve the architecture in response to issues identified, for example:
  - Moving to microservices to address deployment problems.
  - Using cloud computing and containers to address performance and scalability.
- Evolve the pipeline in response to issues identified:
  - Evaluate release patterns to address deployment problems.
  - Engineer the pipeline to make it more robust.
  - Ensure compliance and governance needs are satisfied.

## **Method Used**

This is a hands-on lab-based course where the learning is achieved through applying the practices and techniques in realistic exercises, using common DevOps tools.

### **The technologies used are:**

- A Java project that uses:
  - Ant – build automation.
  - Checkstyle – static code analysis.
  - Junit – unit testing.
  - JaCoCo – test coverage.
  - Cucumber (with Web Driver) – acceptance testing.
  - SoapUI – API testing, also performance and security testing.
- Go CD – continuous integration.
- AWS EC2 – cloud hosting.
- Terraform – provisioning EC2 instances.
- Ansible – configuring EC2 instances and deploying the project.
- Elastic Stack (Logstash, Elasticsearch, Kibana) – log monitoring and analysis.