## **Continuous Delivery**

### **Abstract**

Getting software released to users is often a painful, risky, and time-consuming process. This tutorial sets out the principles and technical practices that enable rapid, incremental delivery of high quality, valuable new functionality to users. Through automation of the build, deployment, and testing process, and improved collaboration between developers, testers and operations, delivery teams can get changes released in a matter of hours–sometimes even minutes–no matter what the size of a project or the complexity of its code base.

In this tutorial we analyze each stage of the delivery process from development through testing to release, discussing at each stage how to improve collaboration and increase feedback so as to make the delivery process as fast and efficient as possible. At the heart of the tutorial is a pattern called the deployment pipeline, which involves the creation of a living system that models your organization's value stream for delivering software. We spend the first half of the tutorial introducing this pattern, and discussing how to incrementally automate the build, test and deployment process, culminating in continuous deployment.

In the second half of the tutorial, we introduce agile infrastructure, including the use of Puppet to automate the management of testing and production environments. We'll discuss automating data management, including migrations. Development practices that enable incremental development and delivery will be covered at length, including a discussion of why branching is inimical to continuous delivery, and how practices such as branch by abstraction and componentization provide superior alternatives that enable large and distributed teams to deliver incrementally.

No laptops are required for the tutorial. It aims to be valuable for everybody involved in software delivery, from testers and developers to managers and systems administrators.

### **Learning outcomes**

\* The importance of fast feedback on the production readiness of your application throughout the delivery process, and how to get it

\* How to improve collaboration between people involved in delivering software (developers, testers, and operations personnel)

\* Principles and practices that enable continuous deployment, including data management

\* How to automate testing and deployment

\* How to evolve your application without using version control branches

\* How to manage continuous delivery with large and distributed teams

\* How to transform your organization's delivery process to enable continuous delivery

### **Morning - 9am-midday**

* Some short case studies of typical delivery environments: ones in which releasing software takes a long time and is painful, and ones which continuously deliver new software. Root cause analysis of slow, risky releases.
* The value proposition of continuous delivery, including the desired end goal.
* The principles of continuous delivery.
* An introduction to the deployment pipeline.

#### **Break - 10:30-10:45**

* The principles and practices of automated functional acceptance testing
* A refresher on continuous integration, with particular reference to how to integrate it with the rest of the delivery process
* Why branching in version control is antithetical to continuous delivery. Using incremental development, a superior alternative to version control branching which allows for application evolution.

#### **Lunch midday - 1pm**

### **Afternoon - 1pm - 4pm**

* Building deployment pipelines for component-based architectures, including dependency management
* Going live: how this differs from deployment. Patterns for continuous deployment.
* Data management and migration

#### Break - 2:20-2:40

* Environment and Infrastructure management, including provisioning, management, and monitoring
* Managing and implementing continuous delivery