Microservices design patterns for Java Applications

Microservices allow to decompose a monolithic application into cohesive and multiple decoupled services. Each service is running in its own process and communicate using lightweight mechanisms, such as HTTP API. These services are built around business capabilities or functional decomposition.

Microservice also enables true polyglot architecture – both in terms of language and data. It truly allows you to employ the right tool for the right job. Each service is independently deployable and lends very well to fully automated deployment machinery.

Can you take an existing Java EE application and decompose it into microservices? What tools are required to enable CI/CD? What are different design patterns for microservices? What tools do you need to manage such services? Is the complexity being pushed around from service implementation to orchestration? This talk will explain some of these concerns and provide guidance on how to leverage microservices in your Java applications.

Arun Gupta is a technology enthusiast, avid runner, author of a best-selling book, globe trotter, a community guy, Java Champion, JavaOne Rockstar, JUG Leader, Minecraft Modder, NetBeans Dream Teamer, Devoxx4Kids-er, and a Red Hatter.

Working With Microservices

We all know that in the real world there is more to developing that writing lines of code. This session will explore how fabric8 has evolved to provide a platform that supports not only the development of Microservices but also working with them, taking an idea from inception right through to running in a live environment.

With popular trends such as DevOps we know that it is more about the culture of an organisation that will give you greater agility and chance of success. Being able to communicate effectively with your cross functional teams increases productivity, reduces social conflicts and establishes the all important feedback loops.

We will look at how fabric8 provides out of the box integration for hosted git services in Gogs, Agile project management with Taiga and social tools such as Lets Chat and Slack to enable intelligent, extendable automation using Hubot, providing a platform that is designed for the new age Microservices team.

We will also cover the integration of popular logging and metric tools that are prerequisites to continuous improvement. We need to understand not only how the platform is operating but also greater visibility of how it's being used.

Being able to visualise how teams communication in and outside of their unit can be seen as first steps to measuring the culture within an organisation. This can be extremely useful in identifying early signs of internal silos developing as well as learning from high performing teams.

James Rawlings, Principal Engineer, Red Hat

James has been a Solutions Architect for the past 2 years at Red Hat, working on DevOps and is a contributor to Fabric8.

Microservices with Apache Camel

Apache Camel is a very popular integration library that works very well with microservice architecture. This talk introduces you to Apache Camel and how you can easily get started with Camel on your computer.

Then we cover how to create new Camel projects from scratch as micro services which you can boot using Camel or Spring Boot, or other micro containers such as Jetty or fat JARs.

We then take a look at what options you have for monitoring and managing your Camel microservices using tooling such as Jolokia, and hawtio web console. The second part of this talk is about running Camel in the cloud.

We start by showing you how you can use the Maven Docker Plugin to create a docker image of your Camel application and run it using docker on a single host. Then kubernetes enters the stage and we take a look at how you can deploy your docker images on a kubernetes cloud platform, and how the fabric8 tooling can make this much easier for the Java developers.

At the end of this talk you will have learned about and seen in practice how to take a Java Camel project from scratch, turn that into a docker image, and how you can deploy those docker images in a scalable cloud platform based on Google's kubernetes.

Claus Ibsen is a principal software engineer from Red Hat.

Claus is working full time as Apache Camel committer.

And is author of the Camel in Action book.

He is also heavily involved with fabric8 and hawtio projects, especially with functionality that involves Camel.

Kubernetes for Java Developers

This talk will give an outline of the Kubernetes open source project from Google, explain how it changes everything from a developers perspective in the hybrid cloud and why its relevant for modern Java developers.

Then we'll outline the key parts of Kubernetes that Java developers need to be aware of and how to adapt your approach to developing, releasing and provisioning Java to work well on premise, on the public or hybrid cloud to deliver value faster to your customers.

James created the Groovy programming language and Apache Camel and was one of the founders of these open source projects: fabric8, hawtio, Apache ActiveMQ, and Apache ServiceMix.

James is currently Senior Consulting Software Engineer at Red Hat focussing on making it easy to work with Kubernetes, OpenShift and Fabric8. James and has more than 20 years of experience in enterprise software development with a background in finance, consulting, and middleware.

Java and Docker

Unless you have been living under a rock, you've probably already heard of Docker. But what exactly does this system-level virtualisation technique offer to Java developers? This talk demonstrates the benefits of using Docker in a development workflow and shows how to integrate Docker into the build process.

The way how we deploy applications will change with Docker. Instead of creating classical WAR and EAR artefacts and deploying them to JEE servers, Docker images now become the central deployment units. These images already contain the execution context and are perfectly suited for new paradigms like Microservices. The Docker image creation can be integrated into the build process in many ways. This talk shows how to use a dedicated Maven plugin for this.

Docker can also help in crafting truly robust, self-contained, isolated and fast integration test suites. We will see how elegantly Docker based integration tests can be setup and run directly from within builds.

Third, since all development tools like Java compiler or Maven are available as Docker images, full reproducible builds across different development and build environments can be easily achieved.

These new concepts are explained in-depth and backed by concrete code examples and live demos. At the end of this talk you will have a good understanding of Docker's benefits from a developer's perspective.

Roland Huß is a Principal Software Engineer working at Red Hat where he is working on Fabric8, an integration and management platform on top of Kubernetes. He has been developing in Java for more than eighteen years but never forgot his roots as system administrator.

Roland is an active open source contributor, lead developer of the JMX-HTTP bridge Jolokia and the most popular docker-maven-plugin.

API Management with Fabric8

The Fabric8 project ships with an HTTP-Gateway to create a single entry point to all Micro Services running in Docker container', hosted by a particular Fabric8 deployment.

On top of the regular gateway features the Fabric8 HTTP-Gateway offers API Management features. The Gateway leverages the asynchronous non-blocking features from the Vert.x project and the API Management features from the APIMan project.

We will demo how to configure and run things like Basic Authentication, Rate Limiting, IP White/Black listing on an 'On Premise Private Cloud' of Raspberry Pi's.

Kurt Stam has been working in the enterprise integration space for the last decade. He has designed and implemented integration solutions for high- volume distributed systems in the telecommunication, financial, and travel industries. He is a core middleware developer at Red Hat and is project lead on the registry and repository projects. He is passionate about open source and holds a Ph.D. in Computational Mechanics.

WF-swarm: does my fatjar look big in this?

Application servers are dead? We hear more and more than Linux containers, microservices and DevOps do away with the need for traditional Java application servers, particularly those based on Java EE.

In this session we'll look at WildFly-swarm a subproject of WildFly, the leading open source Java application server. We'll see how the underlying architecture allows WildFly to be trimmed down to give you Just Enough Application Server (JeAS?) and retain those key capabilities your application (or microservice) needs as well as creating a self-contained executable jar.

Dr. Mark Little serves as the vice president of middleware engineering at Red Hat. Prior to taking over this role in 2008, Mark served as the SOA technical development manager and director of standards. Additionally, Mark was a distinguished engineer and chief architect and co-founder at Arjuna Technologies, a spin-off from HP. He has worked in the area of reliable distributed systems since the mid- 80s with a PhD in fault-tolerant distributed systems, replication, and transactions. Mark and his family reside in Newcastle, UK.

Evolving a Mobile-centric Architecture: The Microservices Way

Red Hat Mobile Application Platform (previously FeedHenry) is an end to end platform for the development, deployment, management and monitoring of mobile-centric solutions.

This talk will start by looking at the evolution of this platform over the last several years - as it transformed from a J2EE monolith into a microservice polyglot based solution. We will look at the design decisions that led to the current architecture and some of the real world problems we have encountered with the microservices approach.

We will then look at microservices from a mobile application perspective. What place do microservices have in a mobile centric world. When, where and how should you use them achieve incredible performance improvements for mobile applications and how can a platform help with this approach?

John Frizelle is the chief architect for the Red Hat Mobile Application Platform. John has been working with mobile since 2008 and with microservice architectures since 2011.