Java Microservices with Spring Cloud: Coordinating Services

INTRODUCING SPRING CLOUD AND MICROSERVICES COORDINATION SCENARIOS

Richard Seroter
SENIOR DIRECTOR OF PRODUCT, PIVOTAL

@rseroter

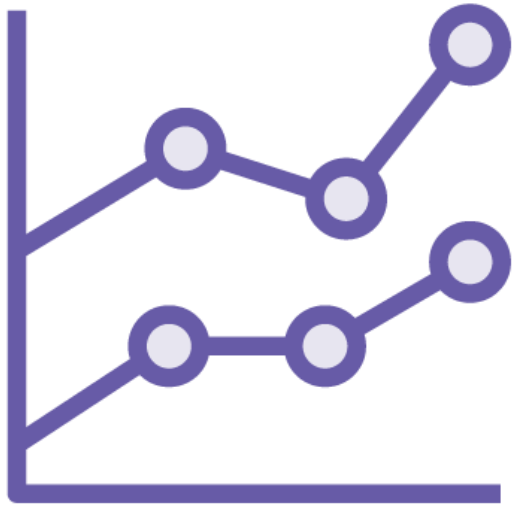
Overview

- Why are microservices architectures so popular?
- Core characteristics of microservices
- Coordination challenges that emerge with microservices
- Microservices with Spring Cloud
- Spring Cloud projects used in this course
- Capabilities we will add in this course
- Goals for the course
- Prerequisites
- Summary

Why Are Microservices Architectures Popular?

Desire for faster changes

Need for greater availability



Motivation for fine-grained scaling

Compatible with a DevOps mindset

Core Characteristics of Microservices

- Components exposed as services
- Tied to a specific domain
- Loosely coupled
- Built to tolerate failure
- Delivered continuously via automation
- Built and run by independent teams

Coordination Challenges that Emerge with Microservices

How do you locate services when hosts change as services get updated or scaled?

How can you dynamically adjust the routing tier?

How do you reduce single points of failure in a distributed architecture?

What can you do to prevent cascading failures when one service starts misbehaving?

Where should you perform load balancing of dynamic services?

What's a good way to introduce loose coupling to an architecture?

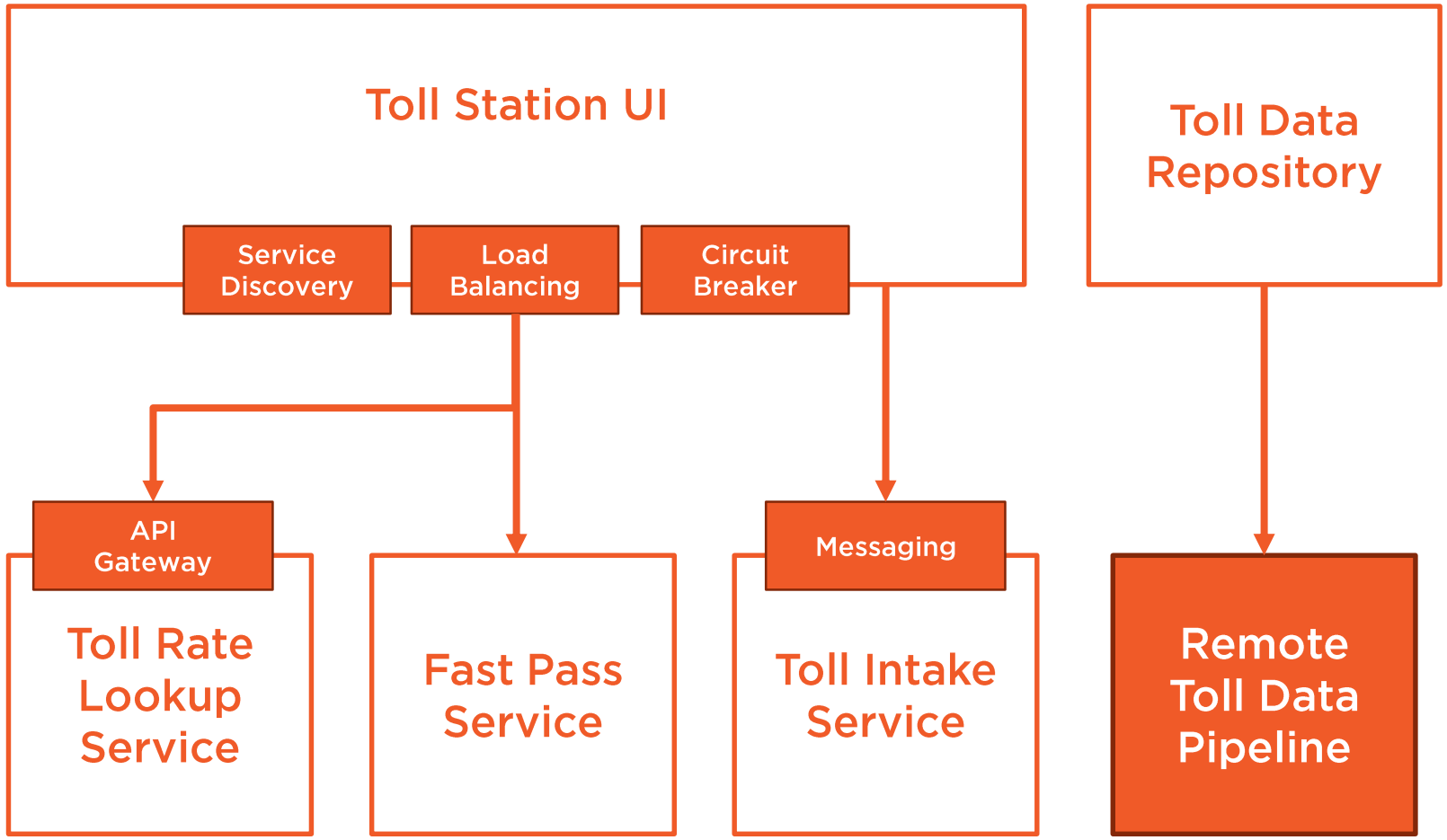
Microservices Scaffolding with Spring Cloud

- Released March 2015
- Build common distributed systems patterns
- Open source software
- Optimized for Spring apps
- Run anywhere
- Includes Netflix OSS technology

Spring Cloud Projects Featured in This Course

- Spring Cloud Eureka
- Spring Cloud Hystrix
- Spring Cloud Ribbon
- Spring Cloud Zuul
- Spring Cloud Stream
- Spring Cloud Data Flow

Capabilities That We Will Add in This Course



Goals for this Course

Recognize challenges and possibilities of coordinating microservices

Get comfortable using leading Spring Cloud projects

Learn how to connect related Spring Cloud projects together

Course Prerequisites

- Base knowledge of Java and OOP
- Familiarity with Spring framework and Spring Boot
- Run a Java-friendly IDE for coding
- Access to a RabbitMQ, MySQL, and Redis environment
- Took previous Java Microservices course (recommended!)

Summary

- Overview
- Why are microservices architectures so popular?
- Core characteristics of microservices
- Coordination challenges that emerge with microservices
- Microservices with Spring Cloud
- Spring Cloud projects used in this course
- Capabilities we will add in this course
- Goals for the course
- Prerequisites