



Spring Boot Apps on Kubernetes

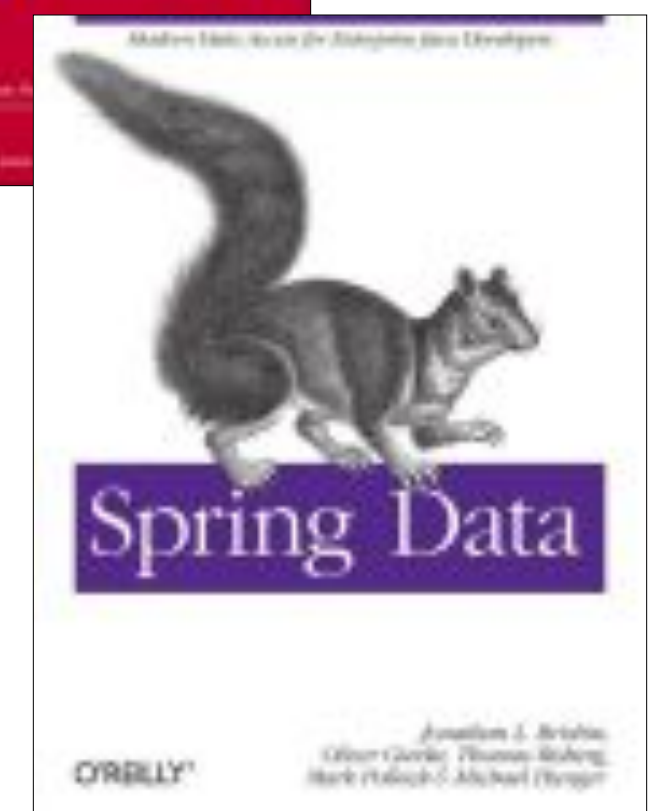


Thomas Risberg
Pivotal
@trisberg

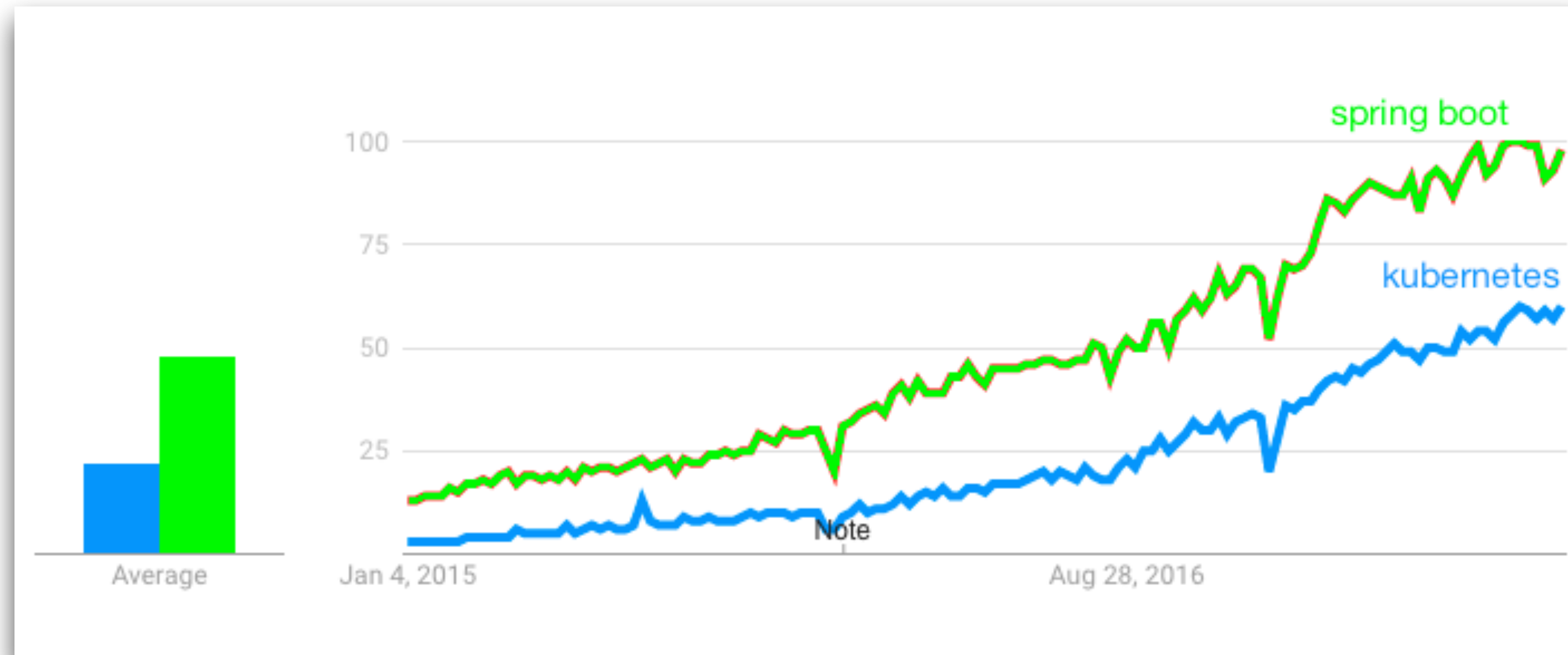
About me

Thomas Risberg (@trisberg)

- Member of the Spring engineering team at Pivotal
- Contributing to *Spring Cloud Data Flow*, *Spring Cloud Deployer for Kubernetes* projects
- Joined the Spring Framework open source project in 2003 working on JDBC support
- co-author of “Professional Java Development with Spring Framework” from Wrox 2005 and “Spring Data” book from O'Reilly 2012



Two Hot Technologies



Based on: <https://trends.google.com/trends/explore?q=kubernetes,spring%20boot>

Press/Analysts

RedMonk
the developer-focus

Videos Research Events

CHARTING STACKS

Language Framework Popularity: A Look at Java, June 2017

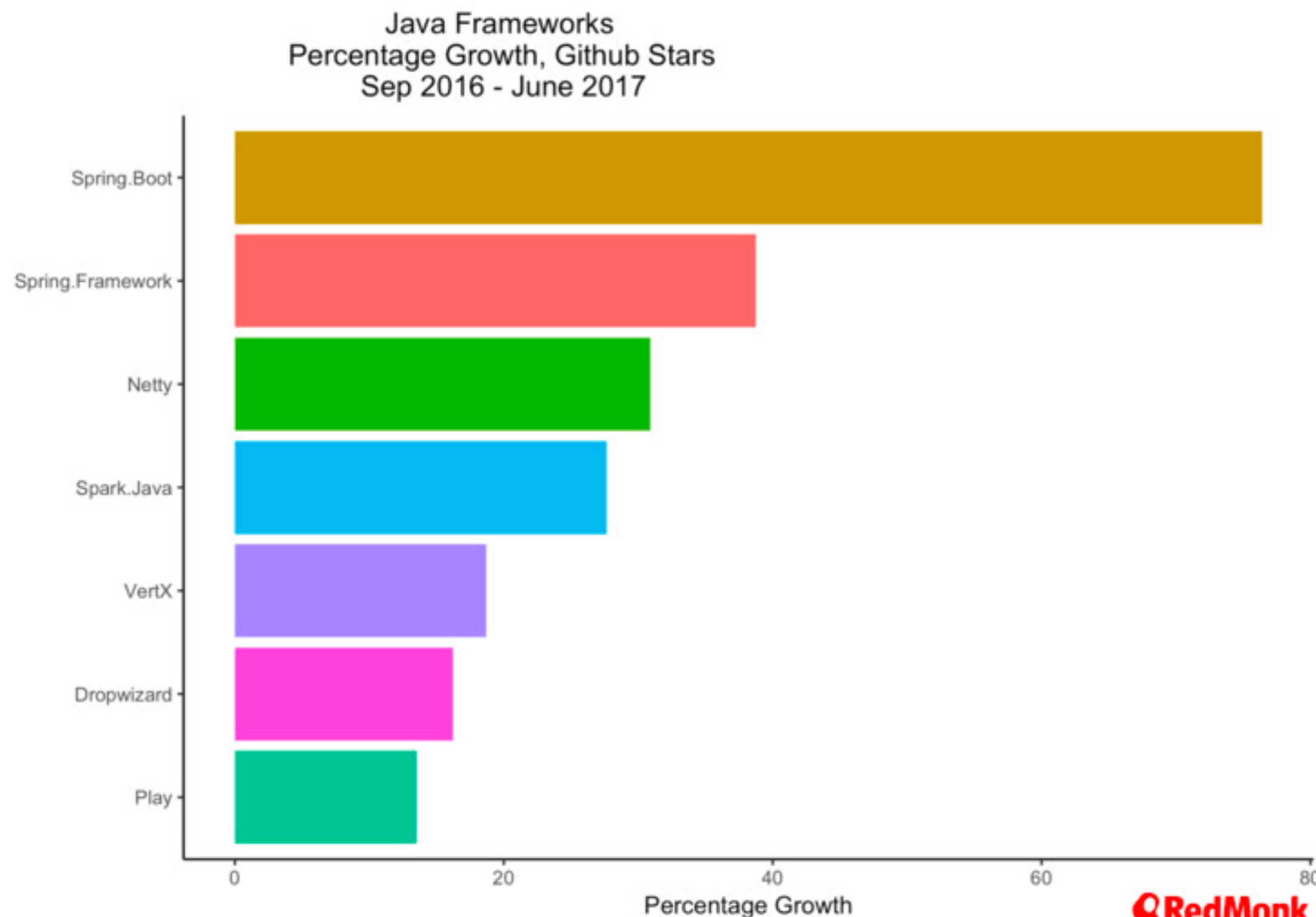
By [fryan](#) | June 22, 2017



TL; DR: Spring Boot is growing at an exponential rate and is set to become the most popular Java Framework soon. Spring Framework and Netty continue to grow strongly

Framework: 'a basic structure'

TL; DR: Spring Boot is growing at an exponential rate and is set to become the most popular Java Framework soon. Spring Framework and Netty continue to grow strongly



techcrunch.com

Kubernetes gains momentum as big-name vendors flock to Cloud Native Computing Foundation

Ron Miller

3-4 minutes



Like a train gaining speed as it leaves the station, the [Cloud Native Computing Foundation](#) is quickly gathering momentum, attracting some of the biggest names in tech. In the last month and a half alone [AWS](#), [Oracle](#), [Microsoft](#), [VMware](#) and [Pivotal](#) have all joined.

What is Spring Boot?

Spring Boot takes an opinionated view of building production-ready Spring applications. Spring Boot favors convention over configuration and is designed to get you up and running as quickly as possible.

<http://projects.spring.io/spring-boot/>

Pair-programming with Spring Team



What is Kubernetes?

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery.

<https://kubernetes.io/>

Pairing with Google SRE Team?

Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.

Running Apps on Minikube

- Build Docker image

```
eval $(minikube docker-env)
docker build -t trisberg/hello:0.0.1 .
```

- Using `kubectl` command line - run app

```
kubectl run hello --image trisberg/hello:0.0.1 --port=8080
```

- Expose the app port

```
kubectl expose deployment hello --type=NodePort
```

Demo

Simple Hello Spring Boot/Kubernetes app deployment

- ▶ <https://github.com/trisberg/boot-k8s/blob/master/demo-hello.adoc>

Spring Cloud / Netflix OSS

- Spring Cloud Config
- Service Discovery
 - Netflix Eureka
 - Consul
- Load balancing / routing
 - Netflix Ribbon & Zuul
- Circuit Breakers
 - Netflix Hystrix

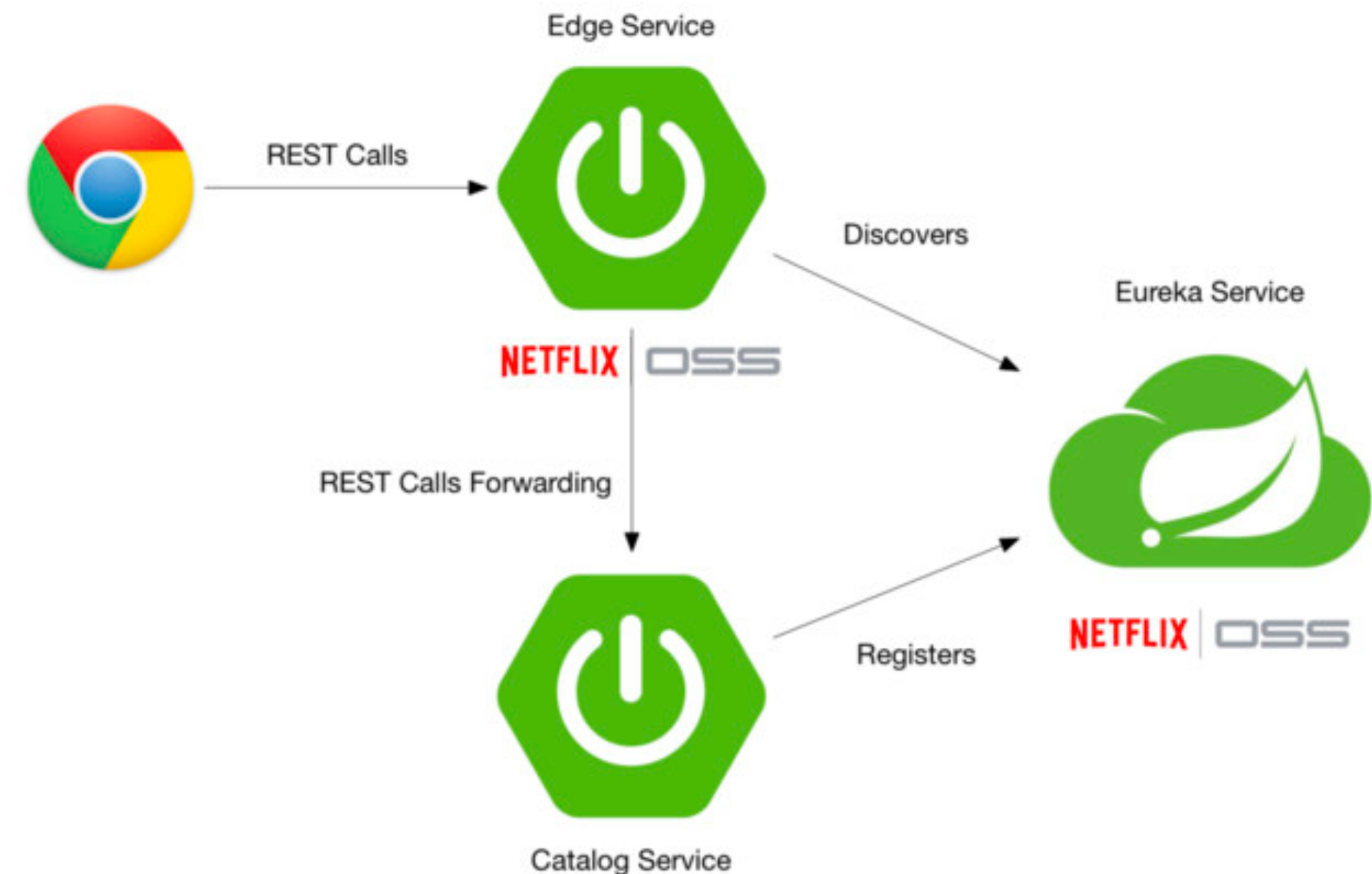
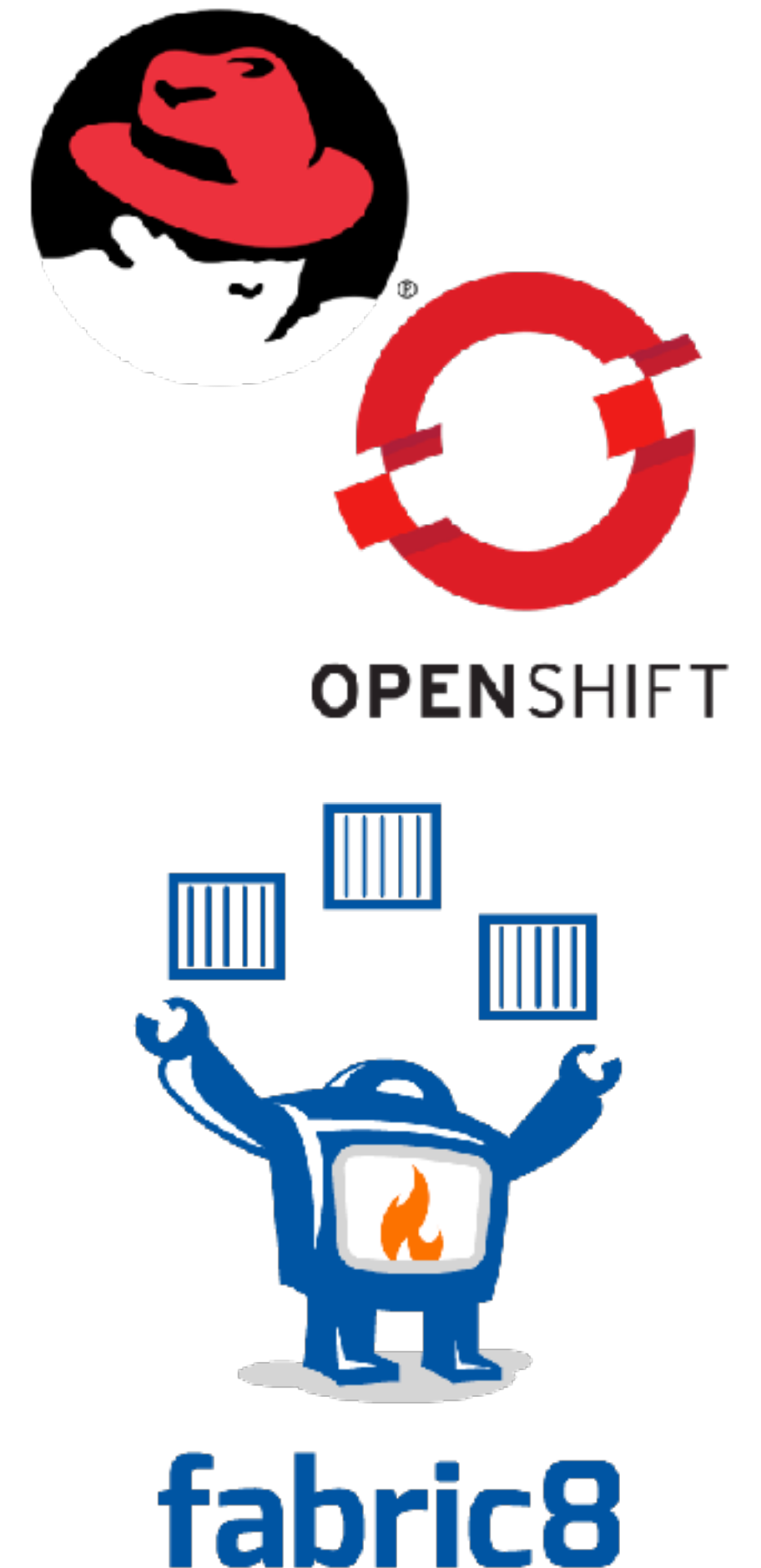


Image from: <https://www.slideshare.net/mraible/develop-hip-apis-and-apps-with-spring-boot-and-angular-connecttech-2017/16?src=clipshare>

Spring Cloud for Kubernetes

- Fabric8 team created some integration libs
spring-cloud-kubernetes
- Now available in spring-cloud-incubator on GitHub



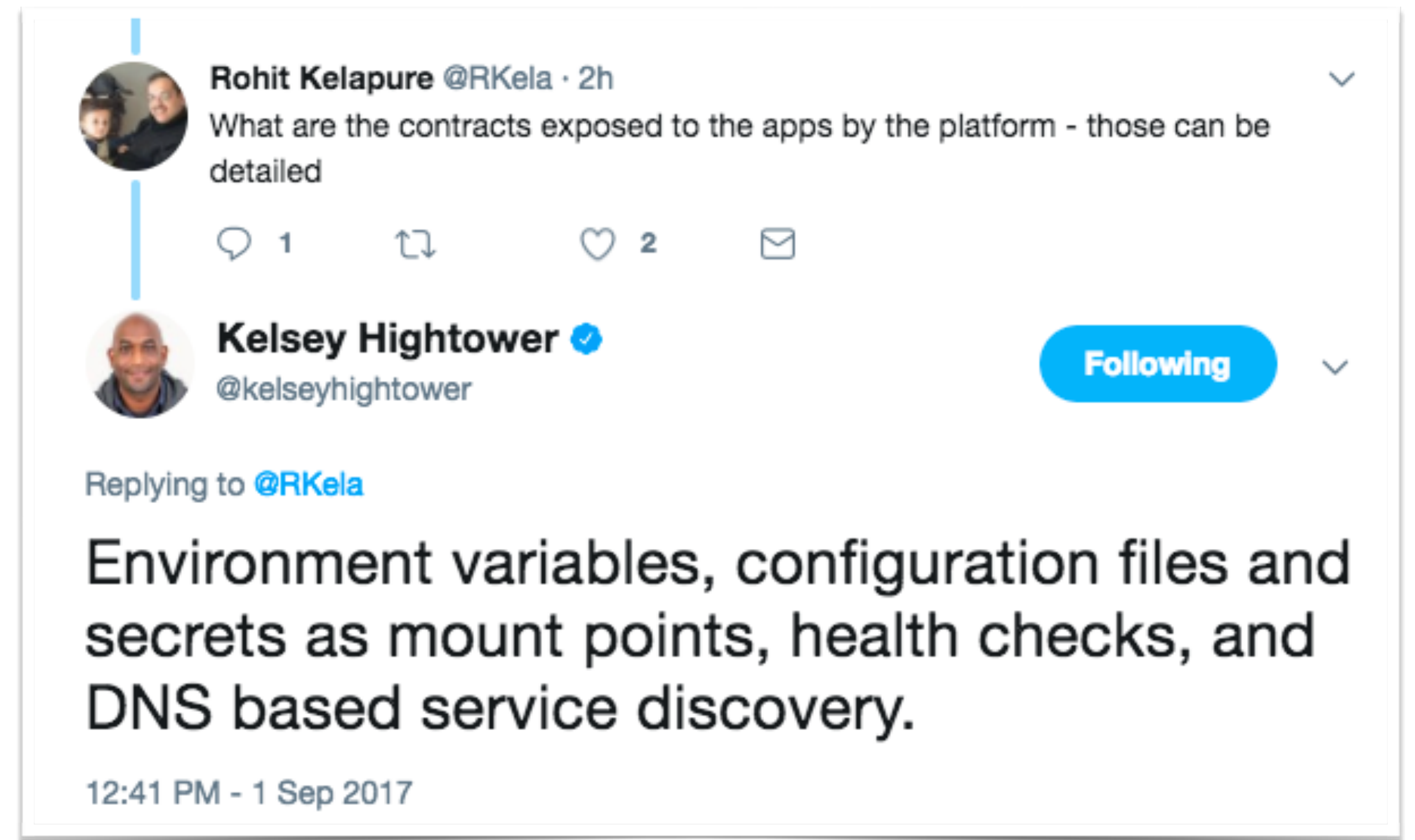
Building Apps for Kubernetes



<https://twitter.com/kelseyhightower/status/903640408613306369>

Contracts exposed to the apps by the platform

- Environment variables
- Configuration files
- Secrets as mount points
- Health checks
- DNS based service discovery



<https://twitter.com/kelseyhightower/status/903643916599046145>

Demo

Simple REST Repository App

- ▶ <https://github.com/trisberg/boot-k8s/blob/master/demo-actors.adoc>

Configuration

```
<dependency>  
  <groupId>org.springframework.cloud</groupId>  
  <artifactId>spring-cloud-starter-kubernetes-config</artifactId>  
  <version>0.2.0.RELEASE</version>  
</dependency>
```

- Environment variables
- Configuration files
- Secrets

```
env:  
- name: SPRING_PROFILES_ACTIVE  
  value: kubernetes  
- name: SPRING_CLOUD_KUBERNETES_CONFIG_NAME  
  value: actors  
- name: SPRING_CLOUD_KUBERNETES_SECRETS_ENABLE_API  
  value: 'true'  
- name: SPRING_CLOUD_KUBERNETES_SECRETS_NAME  
  value: mysql
```


Health Checks

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>
```

```
livenessProbe:  
  httpGet:  
    path: /health  
    port: 80  
  initialDelaySeconds: 90  
  periodSeconds: 15  
  timeoutSeconds: 5  
readinessProbe:  
  httpGet:  
    path: /health  
    port: 80  
  initialDelaySeconds: 45  
  periodSeconds: 15  
  timeoutSeconds: 5
```

```
resources:  
  limits:  
    cpu: 1.0  
    memory: 1024Mi  
  requests:  
    cpu: 0.5  
    memory: 640Mi
```

Service Discovery

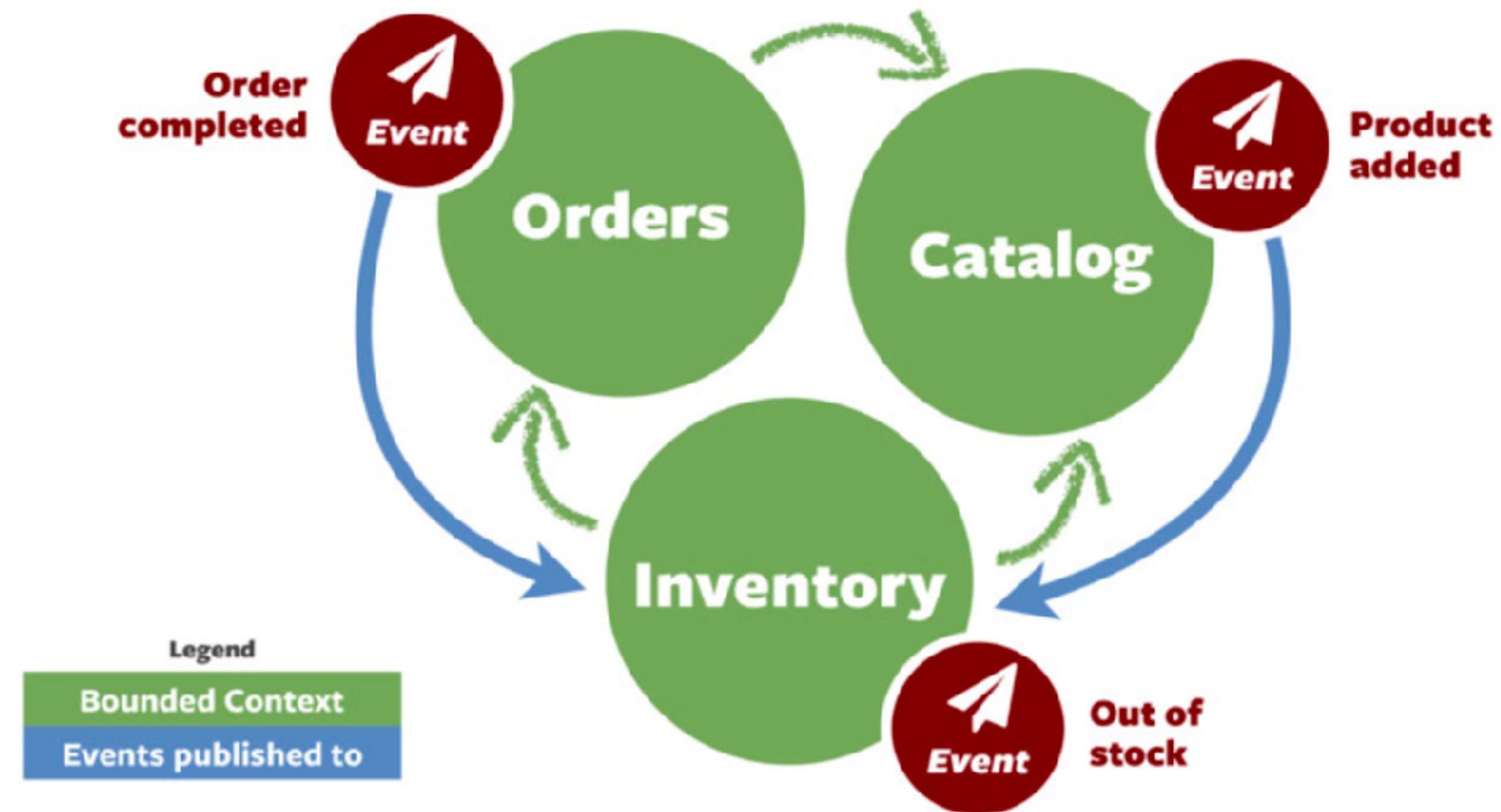
ConfigMap using env vars:

```
data:
  application.yaml: |-
    security:
      basic:
        enabled: false
    spring:
      datasource:
        url: jdbc:mysql://${MYSQL_SERVICE_HOST}:${MYSQL_SERVICE_PORT}/mysql
        username: root
        password: ${mysql-root-password}
        driverClassName: com.mysql.jdbc.Driver
        testOnBorrow: true
        validationQuery: "SELECT 1"
```

or DNS: `url: jdbc:mysql://mysql:3306/mysql`

Microservice Architecture Concerns

- Externalized Configuration
 - ConfigMap and Secrets
- Service Discovery
 - DNS, DiscoveryClient
- Circuit-breaker
- Distributed Tracing
- Metrics
- Log aggregation



Circuit-breaker

Netflix Hystrix

```
@HystrixCommand(commandKey = "ActorByName", fallbackMethod = "noActors", commandProperties = {
    @HystrixProperty(name = "execution.isolation.thread.timeoutInMilliseconds", value = "5000")
})
Collection<Resource<ActorResource>> getActorByName(@PathVariable String name) {

    ...

    return actors;
}

Collection<Resource<ActorResource>> noActors(@PathVariable String name) {

    ...

    return empty;
}
```

```
@SpringBootApplication
@EnableCircuitBreaker
public class GatewayApplication {

    public static void main(String[] args) {
        SpringApplication.run(GatewayApplication.class, args);
    }
}
```

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-hystrix</artifactId>
</dependency>
```


Distributed Tracing

Zipkin

```
<dependency>  
  <groupId>org.springframework.cloud</groupId>  
  <artifactId>spring-cloud-starter-kubernetes-zipkin</artifactId>  
  <version>0.2.0.RELEASE</version>  
</dependency>
```

```
spring.sleuth.sampler.percentage=1.0
```

```
containers:  
- name: zipkin  
  image: docker.io/openzipkin/zipkin:latest  
  imagePullPolicy: IfNotPresent  
  ports:  
  - containerPort: 9411  
  env:  
  - name: POD_NAMESPACE  
    valueFrom:  
      fieldRef:  
        apiVersion: v1  
        fieldPath: metadata.namespace
```

Metrics

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>
```

```
<dependency>  
  <groupId>com.moelholm</groupId>  
  <artifactId>prometheus-spring-boot-starter</artifactId>  
  <version>1.0.2</version>  
</dependency>
```

- /metrics
- /prometheus

Pizza Maturity Model

