

Spring Cloud Services on Pivotal Cloud Foundry

Frameworks To Support Cloud Native

Spring

#1 Enterprise Java
App Framework

Netflix

Open Source

Spring Boot

Quick Start

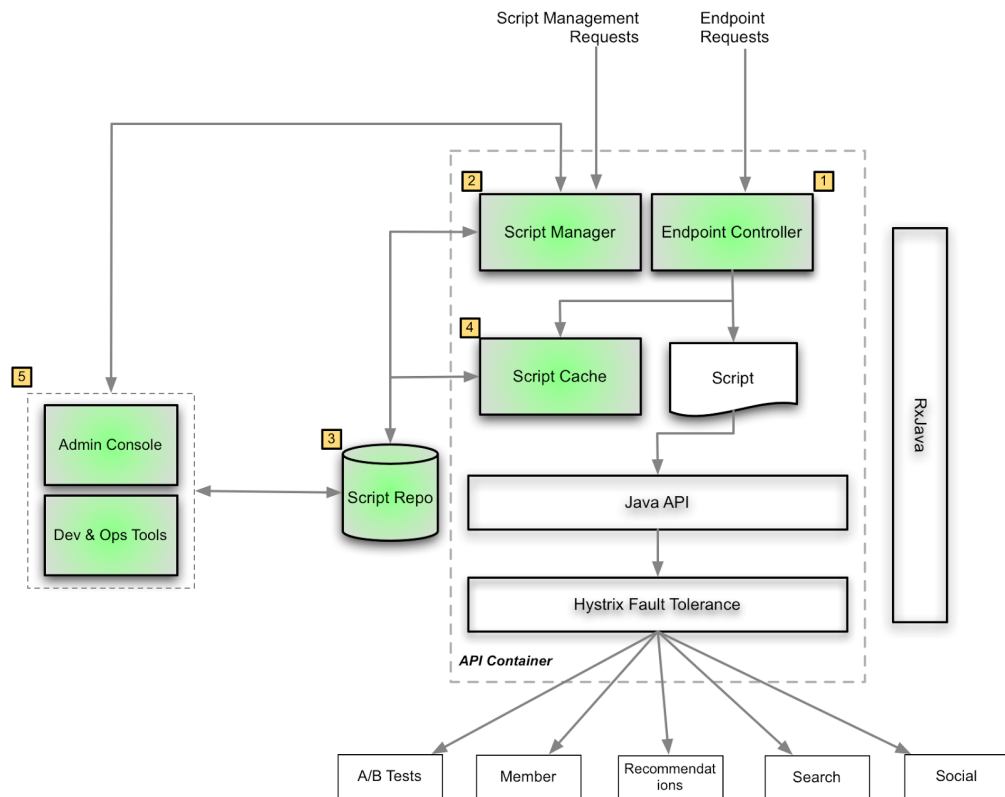
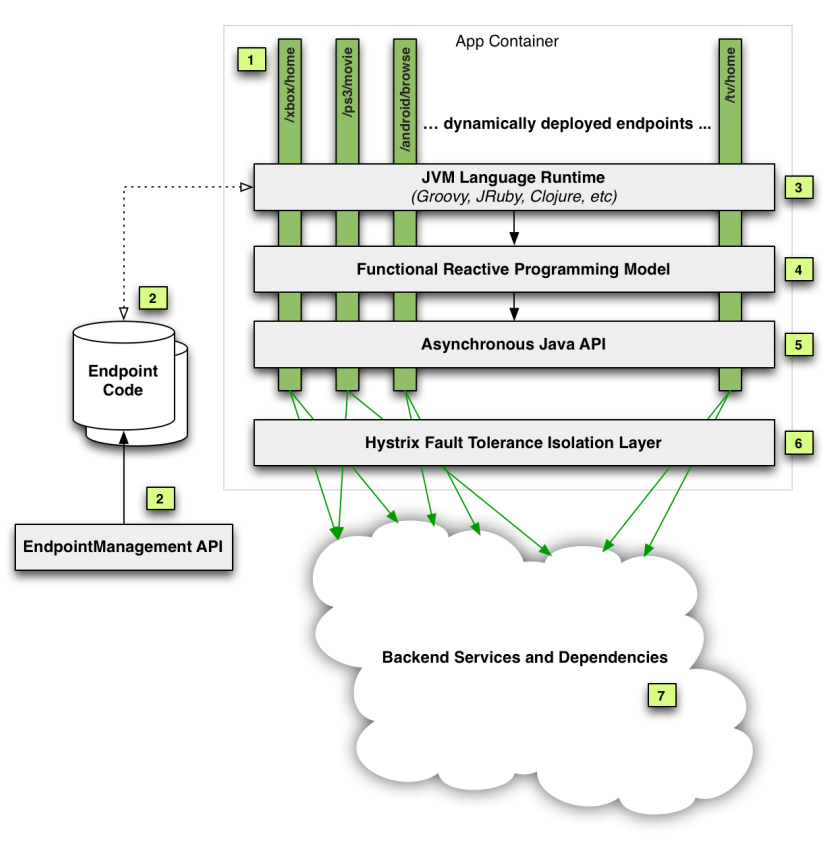
Little to no config

Opinionated View

Spring Cloud

Distributed Config
Service Registration
and Discovery
Routing/Load
Balancing
Service Integration
Fault Tolerance

Netflix Invented A Lot Of Hard Stuff



<http://techblog.netflix.com/2013/01/optimizing-netflix-api.html>

<http://techblog.netflix.com/2015/03/the-netflix-dynamic-scripting-platform.html>

NETFLIX

OSS

- Eureka (Service Discovery)
- Hystrix + Turbine (Circuit Breaker)
- Ribbon (Load Balancer)
- Feign (REST Client)
- Zuul (Gateway/Proxy)

Spring Cloud And Netflix OSS

- Tools to solve common problems in distributed systems
- Spring Cloud Config Server (distributed configuration)
- Eureka (service registration and discovery)
- Hystrix (circuit breaker/fault tolerance library)
- Hystrix Dashboard (service health dashboard)
- Ribbon (client side load balancing, including RestTemplate integration)
- Feign (easy REST clients)
- Zuul (routing & server side load balancing)
- Spring Cloud Bus (distributed Spring Boot actuator)
- Spring Cloud Security

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

Spring Cloud Services Components



Spring Cloud Services



Config Server



Service Registry



**Circuit Breaker
Dashboard**

Spring Cloud Services Overview

Enabling μ Services on the World's Leading Spring Platform: Pivotal Cloud Foundry

CONFIG SERVER

- Externalized application configuration management to easily manage external properties for applications across all environments
- Pivotal Developed

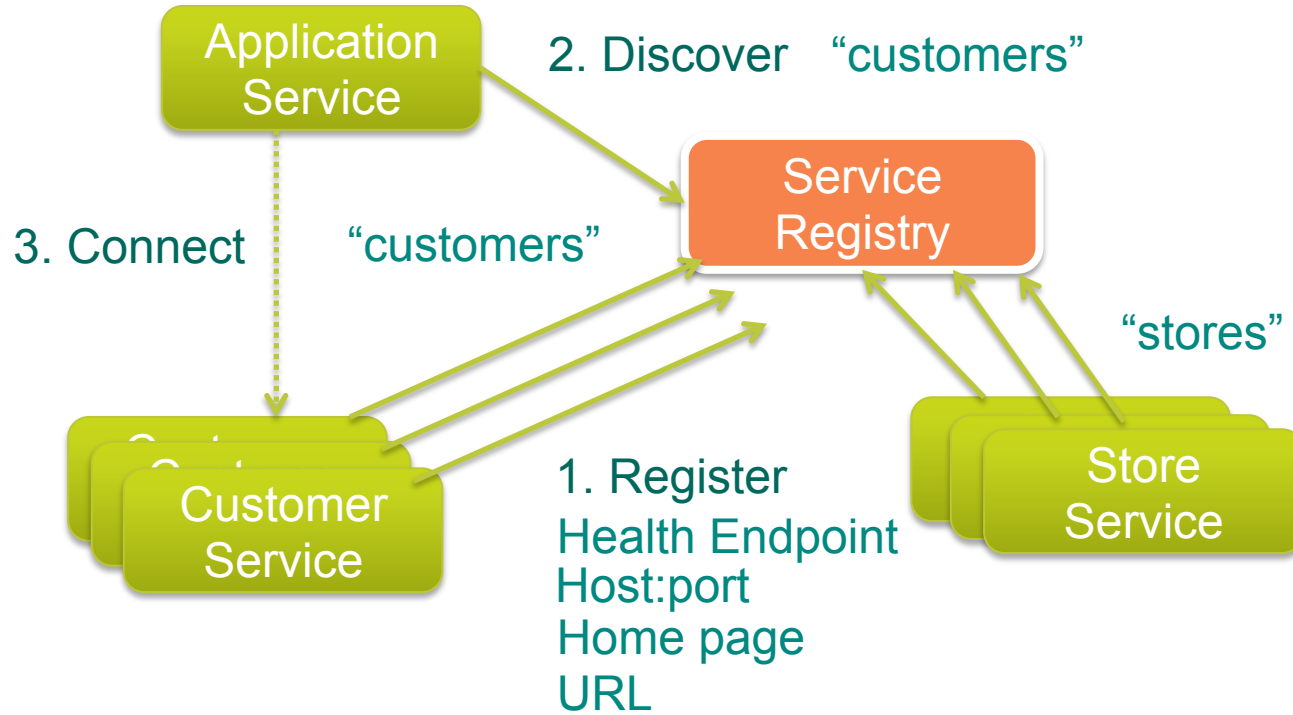
SERVICE REGISTRY

- Enable service discoverability for applications running in a Pivotal Cloud Foundry environment
- Netflix OSS: Eureka

CIRCUIT BREAKER

- Spring applications running in PCF can utilize the Circuit Breaker Pattern
- Netflix OSS: Hystrix and Turbine Circuit Breaker

Service Discovery (Eureka)




Service Discovery: Running a Registry

```
@SpringBootApplication
@EnableEurekaServer
public class EurekaApplication {

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

}
```



Just a Spring Boot Application

Service Discovery: Registration/Discovery

```
@SpringBootApplication
@EnableCircuitBreaker
@EnableDiscoveryClient ←
public class CustomerApp extends RepositoryRestMvcConfiguration {

    @Override
    protected void configureRepositoryRestConfiguration(RepositoryRestConfiguration
config) {
        config.exposeIdsFor(Customer.class);
    }

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

Spring Cloud Service Registry



Service Registry

- Service Registration and Discovery via Netflix OSS Eureka
- Service Binding via Spring Cloud Connector
- Single-tenant, scoped to CF space
- Registration via CF Route

Spring Cloud Service Registry Dashboard



Service Registry for Pivotal CF

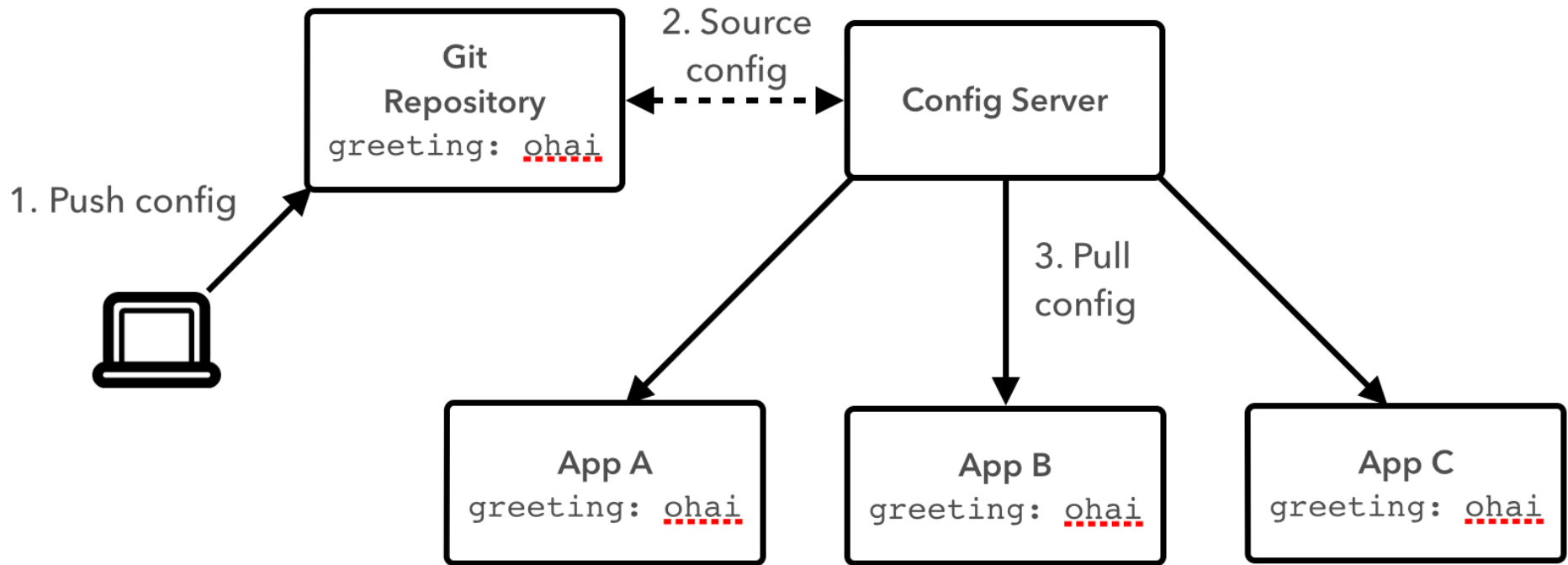
[Home](#) [History](#)

Service Registry Status

Registered Apps

Application	Availability Zones	Status
EUREKA-SAMPLE	default (5)	<div><div>DOWN (1)</div><ul style="list-style-type: none">eureka-sample.deer.wild.cf-app.com:eureka-sample:3f02c70260acf3a47b25c9d08d51e834<div>UP (4)</div><ul style="list-style-type: none">eureka-sample.deer.wild.cf-app.com:eureka-sample:3f02c70260acf3a47b25c9d08d51e834</div>

Spring Cloud Config Server



Spring Cloud Config Server

```
@SpringBootApplication
@EnableConfigServer ←
public class ConfigServerApplication {

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

}
```

Just a Spring Boot Application


Spring Cloud Config Server



Config Server

- Service Binding via Spring Cloud Connector
- Git/SVN URL for Config Repo provided via Service Dashboard (post-provisioning)
- Single tenant, scoped to CF space (nothing prevents shared Git repo)

Spring Cloud Config Server Dashboard

 **Spring Cloud Services** for Pivotal CF

Config Server

Instance ID: 2c320caf-af46-4258-8e5d-b7e97aa979c0

Configuration Source

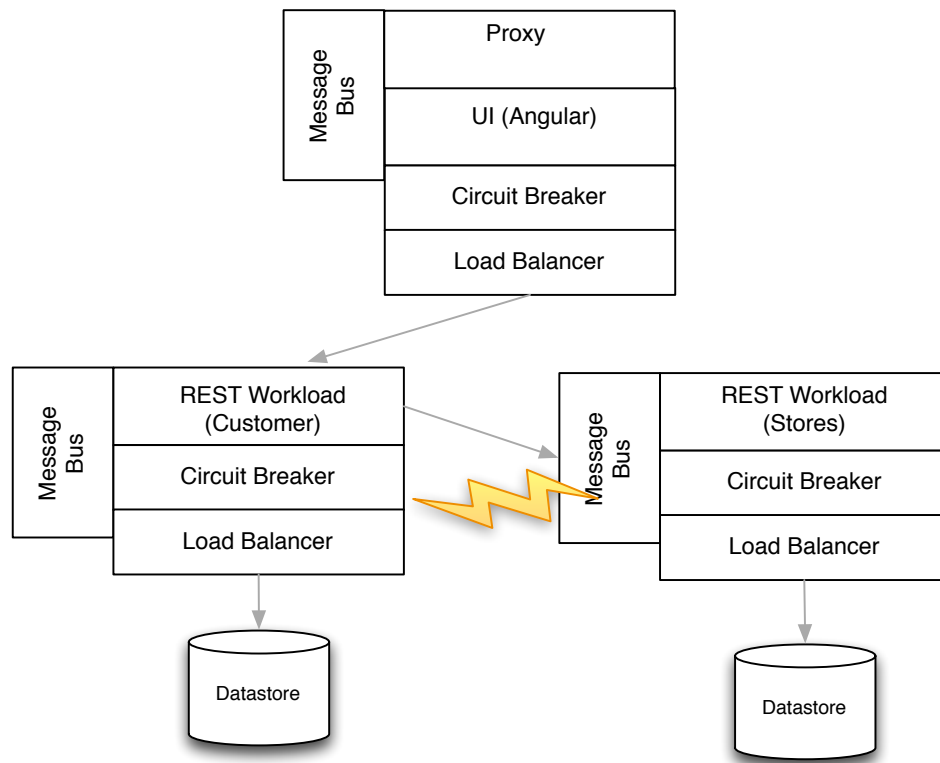
☒ Git
☐ Local

Git URI

Submit


Pivotal © 2015 Pivotal Software Inc. All rights reserved.

Circuit Breaker Example



Enable Circuit Breaker

```
@HystrixCommand(fallbackMethod = "defaultLink")  
public Link getStoresByLocationLink(Map<String, Object> parameters) {  
    URI storesUri = URI.create(uri);  
    try {  
        ServiceInstance instance = loadBalancer.choose("stores");  
        storesUri = URI.create(String.format("http://%s:%s",  
instance.getHost(), instance.getPort()));  
    }  
    catch (RuntimeException e) {  
        // Eureka not available  
    }  
  
    Traversal traversal = new Traversal(storesUri, MediaType.HAL_JSON);  
    Link link = traversal.follow("stores", "search", "by-location")  
        .withTemplateParameters(parameters).asLink();  
  
    return link;  
}
```



Client-Side Load Balancing

Enable Circuit Breaker

```
@SpringBootApplication
@EnableCircuitBreaker ←
@EnableDiscoveryClient
public class CustomerApp extends RepositoryRestMvcConfiguration {

    @Override
    protected void configureRepositoryRestConfiguration(RepositoryRestConfiguration
config) {
        config.exposeIdsFor(Customer.class);
    }

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

Enable Circuit Breaker – Fallback

```
@HystrixCommand(fallbackMethod = "defaultLink")
    public Link getStoresByLocationLink(Map<String, Object> parameters) {
//...
    }
```

```
public Link defaultLink(Map<String, Object> parameters) {
    return null;
}
```



Spring Cloud Services Circuit Breaker Dashboard



**Circuit Breaker
Dashboard**

- Netflix OSS Turbine + Hystrix Dashboard
- Aggregation via AMQP (RabbitMQ)
- Binding via Spring Cloud Connector
- Single-tenant, scoped to CF space

Pivotal

A NEW PLATFORM FOR A NEW ERA