Construindo Microservices Auto-curáveis com Spring Cloud e Netflix OSS

Rodrigo Cândido da Silva
@rcandidosilva

About Me

- Software Architect
  - http://integritastech.com
- JUG Leader do GUJavaSC
  - http://gujavasc.org
- Twitter
  - @rcandidosilva
- Contatos
  - http://rodrigocandido.me

GUJavaSC

Grupo de Usuários Java de Santa Catarina

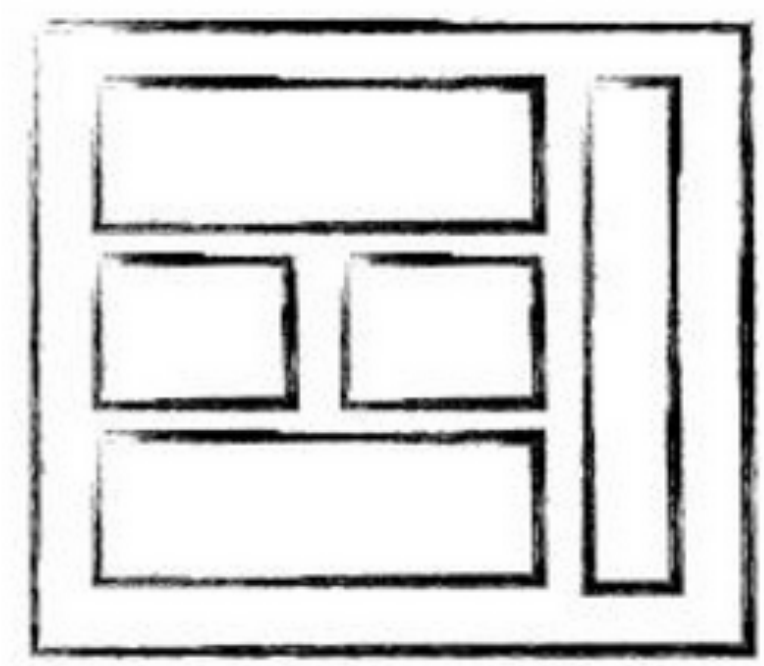
Agenda

- Monolito vs. Microservices
- Principais Desafios
- Spring Cloud + Netflix OOS
  - Spring Cloud Config + Bus
  - Netflix Eureka
  - Netflix Ribbon
  - Netflix Hystrix + Turbine
  - Netflix Zuul
  - Spring Cloud Security
- Conclusões
- Perguntas

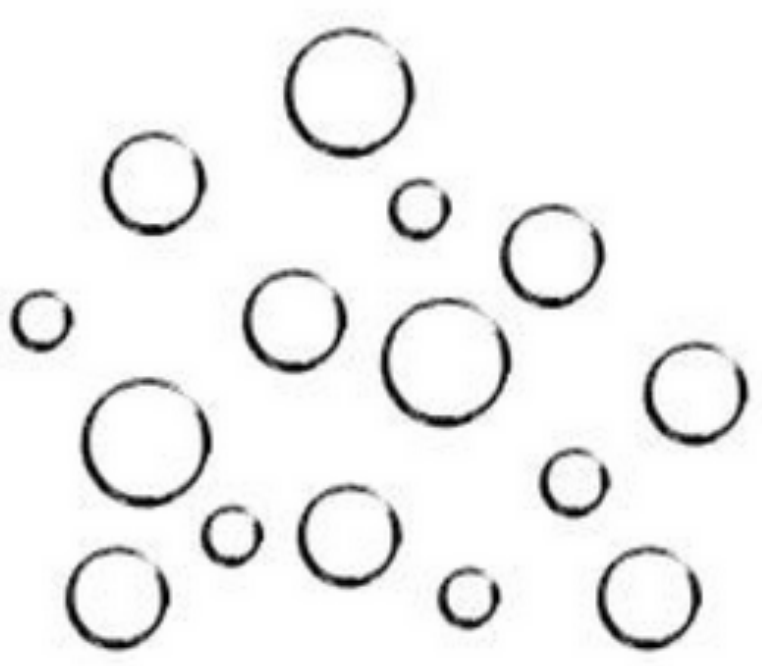
Monolito vs. Microservices

1990s and earlier	2000s	2010s
Coupling		
Pre-SOA (monolithic) Tight coupling	Traditional SOA Looser coupling	Microservices Decoupled

Monolito vs. Microservices



MONOLITHIC/LAYERED



MICRO SERVICES

Microservices

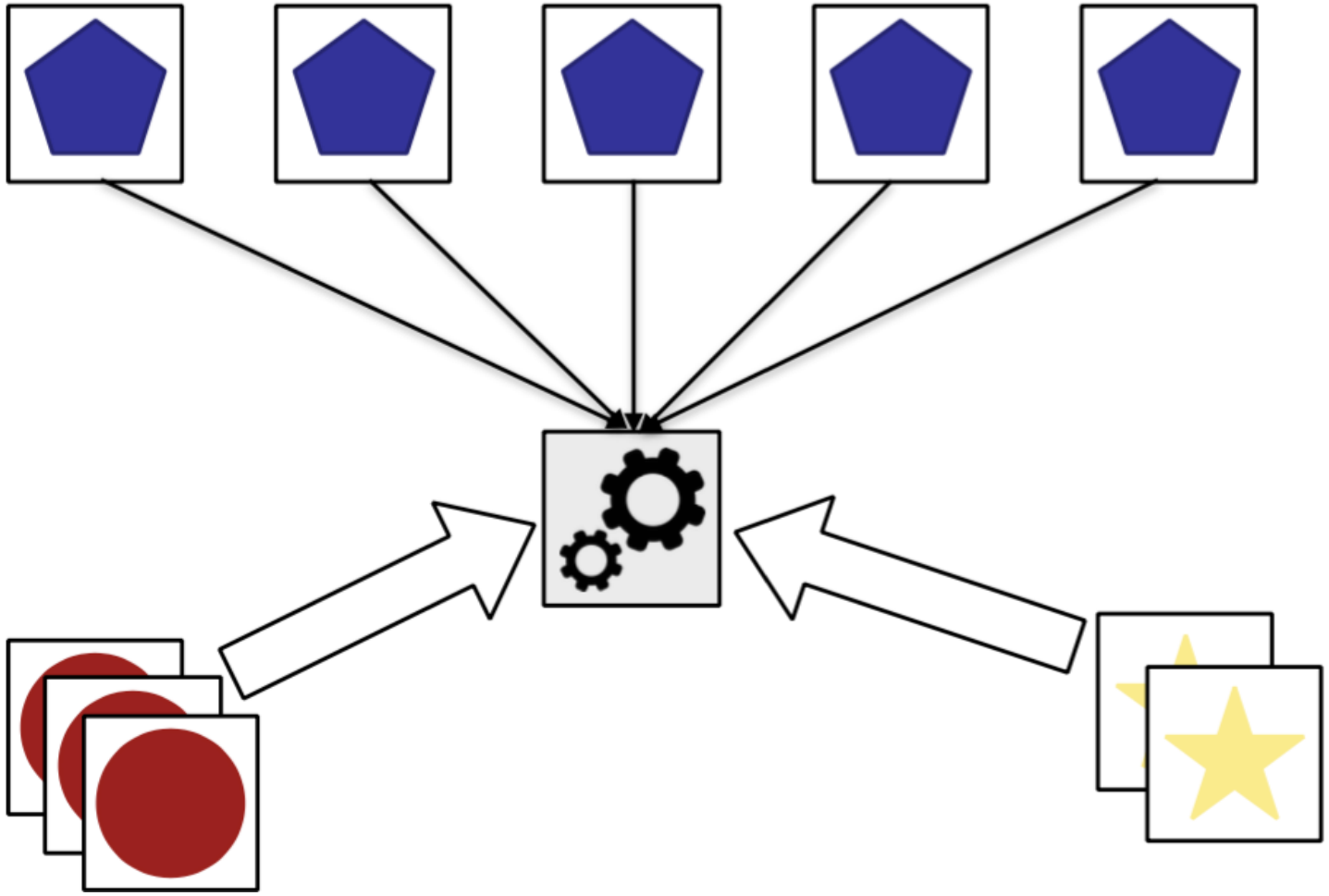
"Small independent component with **well-defined boundaries** that's **doing one thing,** but doing it well**"**

- **Características**
  - Pequenos
  - Deployment interdependentes
  - Independente de tecnologia
  - Independente de infra-estrutura

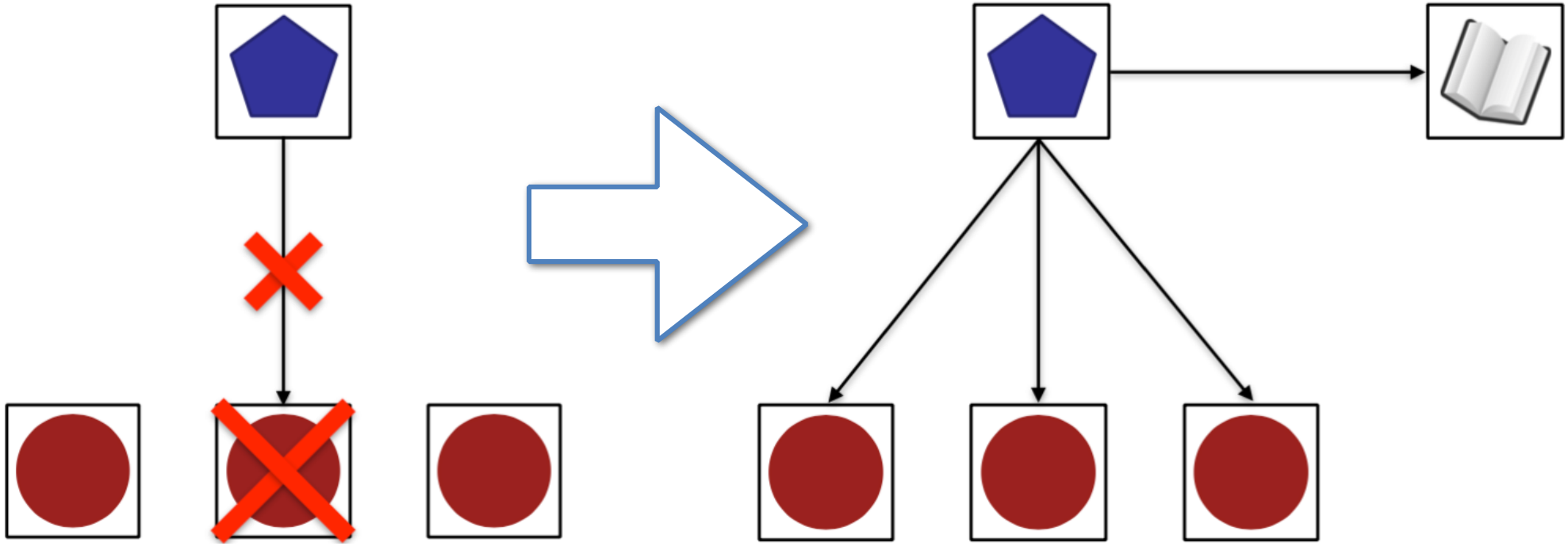
Microservices

- **Como torná-los auto-curáveis?**
  - Gerenciamento de configuração
  - Registro e descoberta dos serviços
  - Roteamento
  - Balanceamento de carga
  - Tolerância à falhas
  - Monitoramento

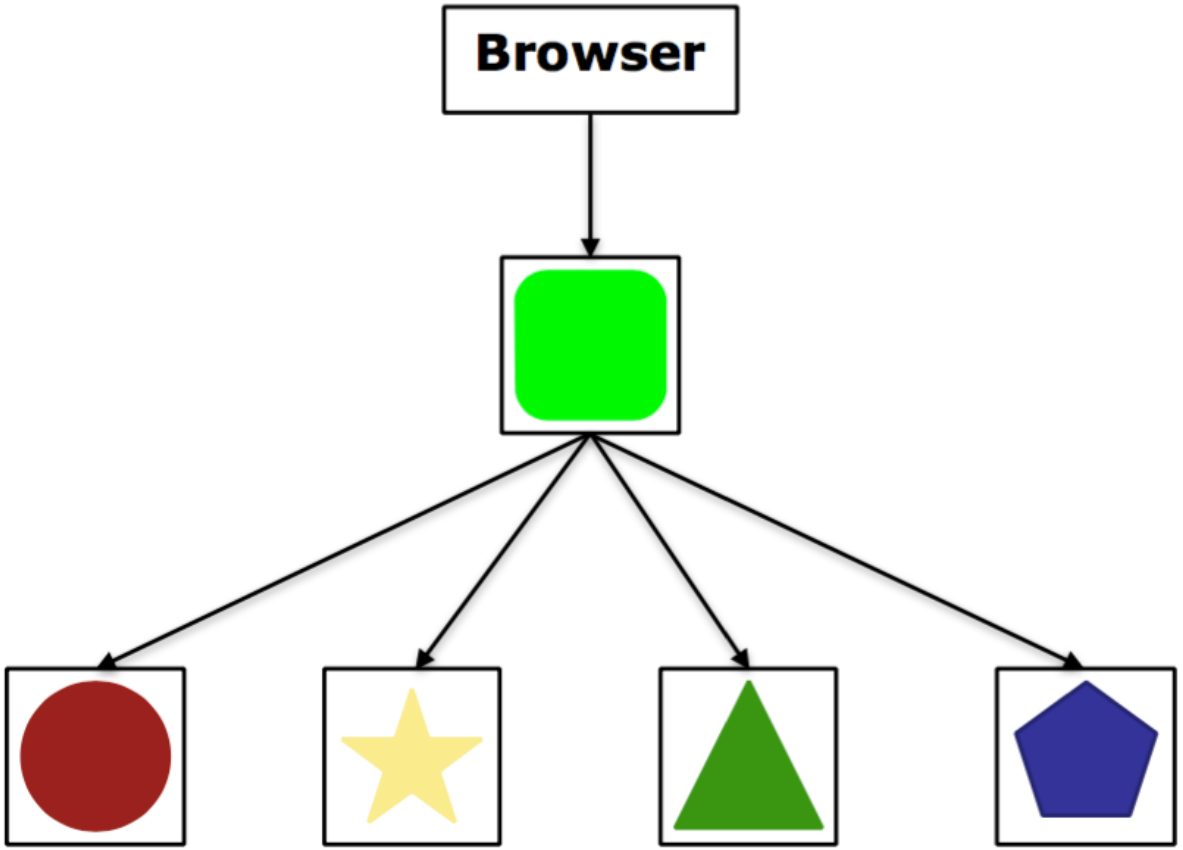
Gerenciamento de Configuração



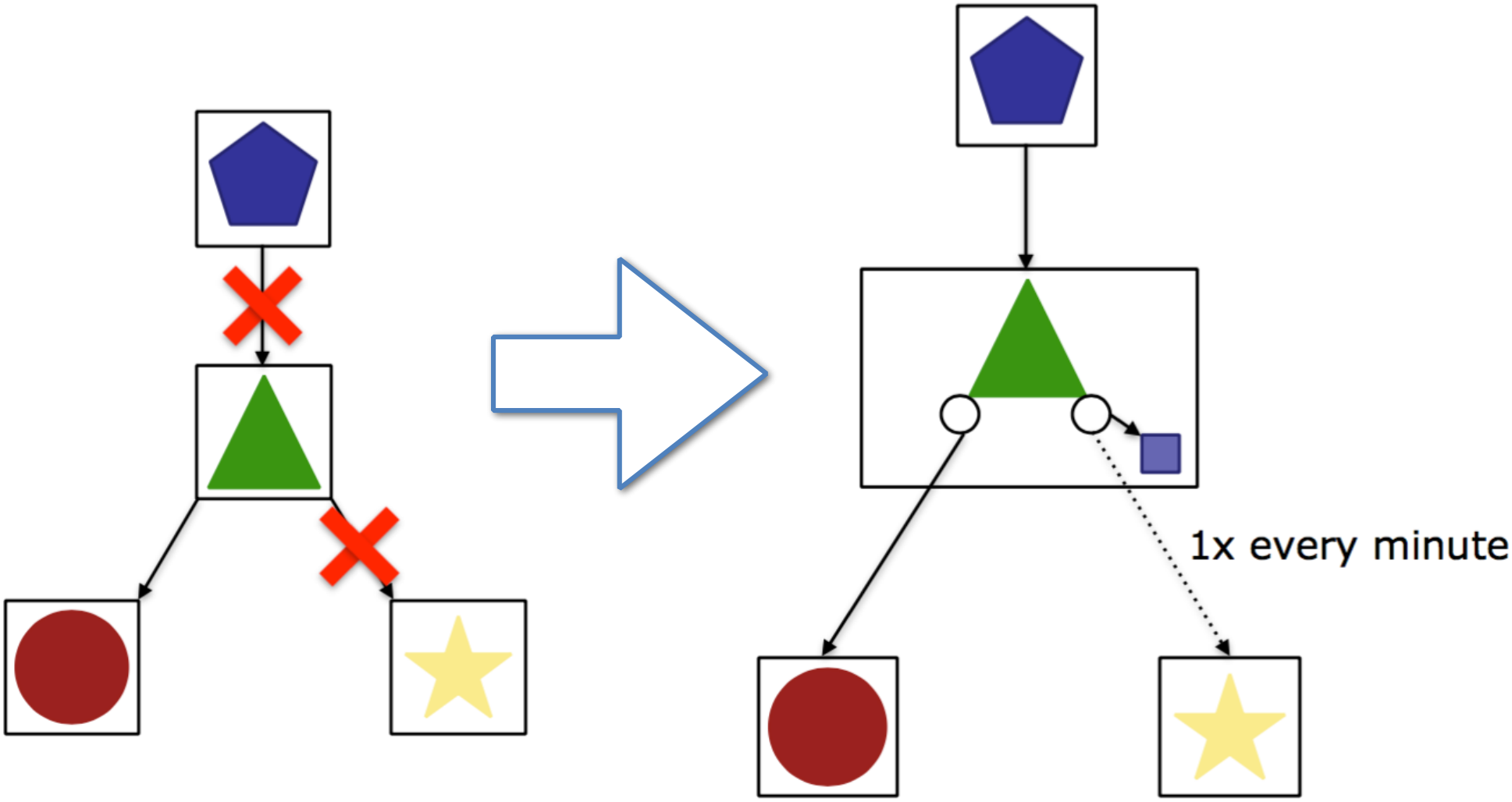
Registro e Descoberta de Serviços



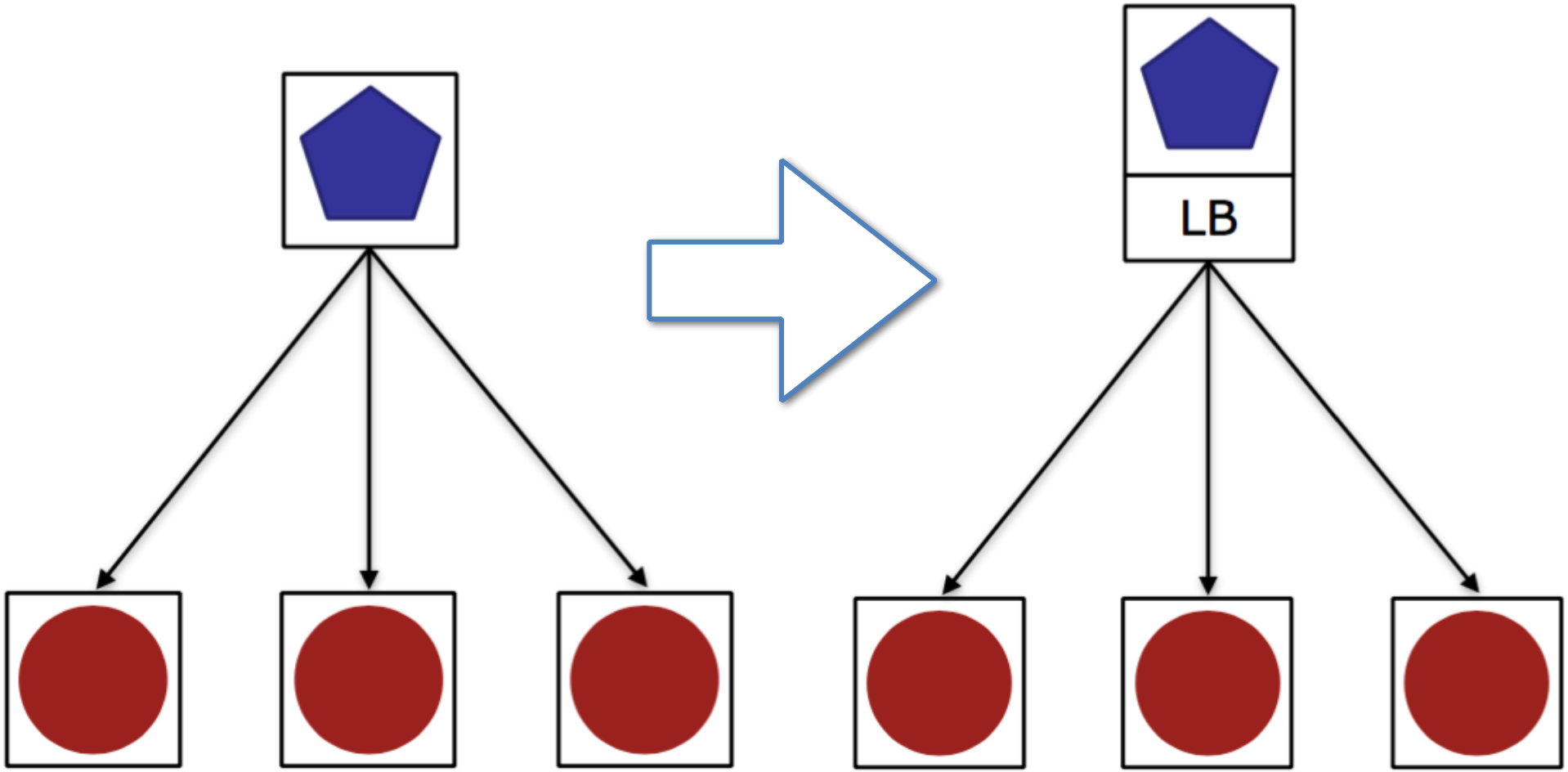
Roteamento



Tolerância à Falhas



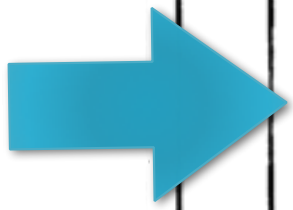
Balanceamento de Carga



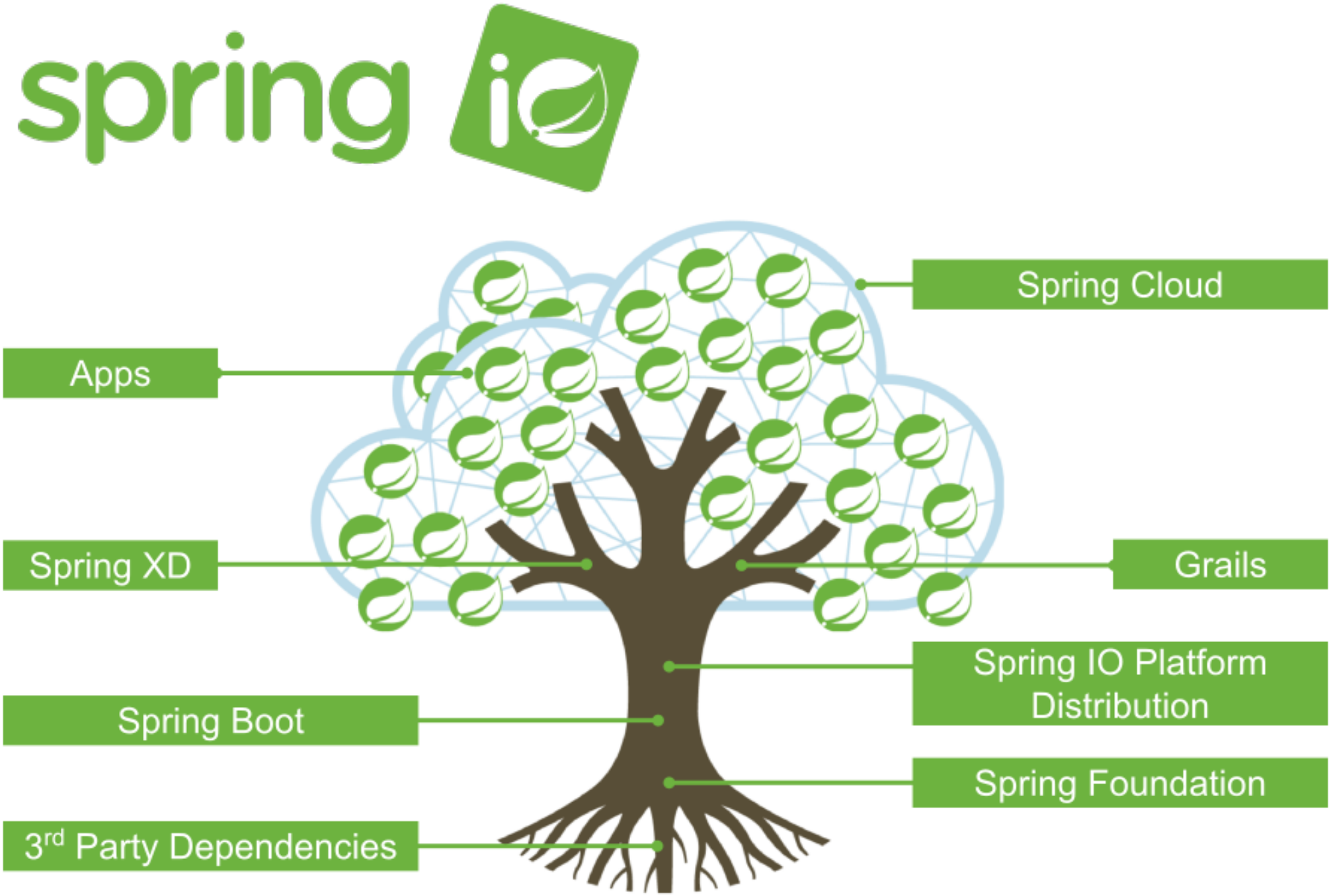
NETFLIX

NETFLIX OSS

- API
- Routing / Health check
- Microservices
- Logging
- Data Management



- Eureka
- Hystrix + Turbine
- Ribbon
- Zuul
- + alguns outros…


spring io
Spring Cloud
Apps
Spring XD
Grails
Spring IO Platform
Distribution
Spring Boot
Spring Foundation
3rd Party Dependencies

Spring Cloud

“Toolset designed for building distributed systems”

- Conjunto de bibliotecas / componentes
  - Não é apenas uma ferramenta
- Integrado ao Spring Boot
- Suporta diferentes arquiteturas e tecnologias em Cloud
  - *AWS, Netflix, Heroku, Cloud Foundry, etc*
- Facilita a implementação de padrões necessários aos sistemas distribuídos

Spring Cloud

- **Principais Componentes**

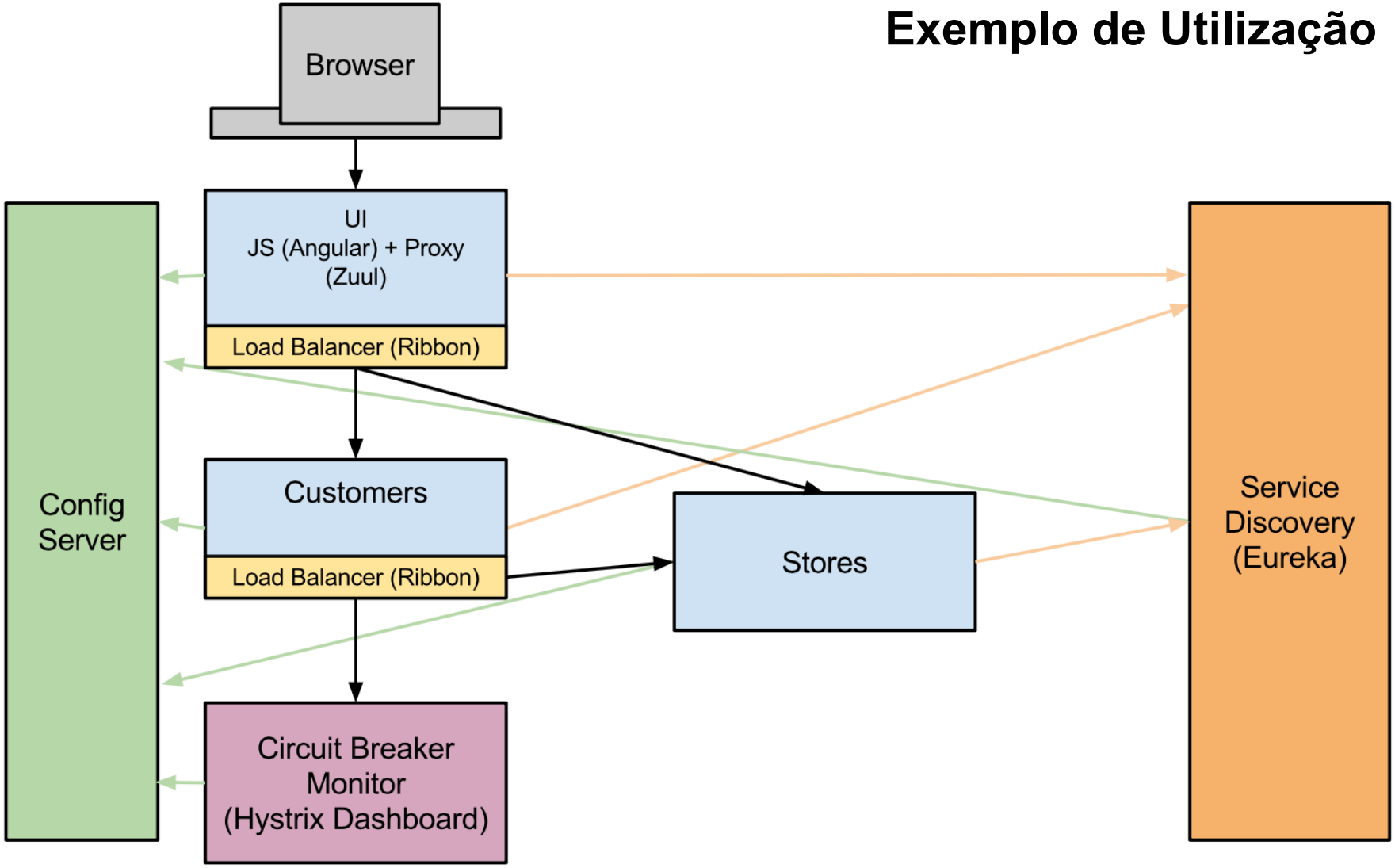
Spring Cloud + Netflix OSS

"Casamento perfeito para criação de microservices auto-curáveis"

Gerenciamento de configuração	***Spring Cloud Config + Bus***
Descoberta de serviços	***Netflix Eureka***
Balanceamento de carga	***Netflix Ribbon***
Tolerância à falhas	***Netflix Hystrix + Turbine***
Roteamento	***Netflix Zuul***
Segurança	***Spring Cloud Security***

Spring Cloud + Netflix OSS

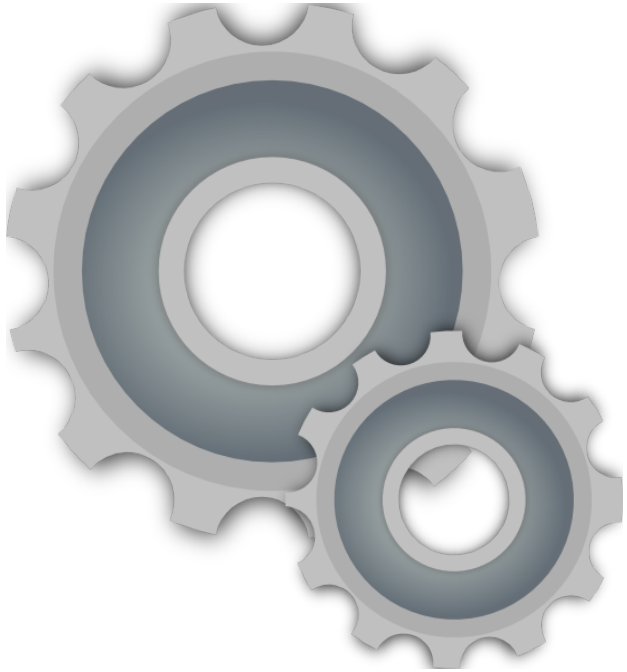
Exemplo de Utilização



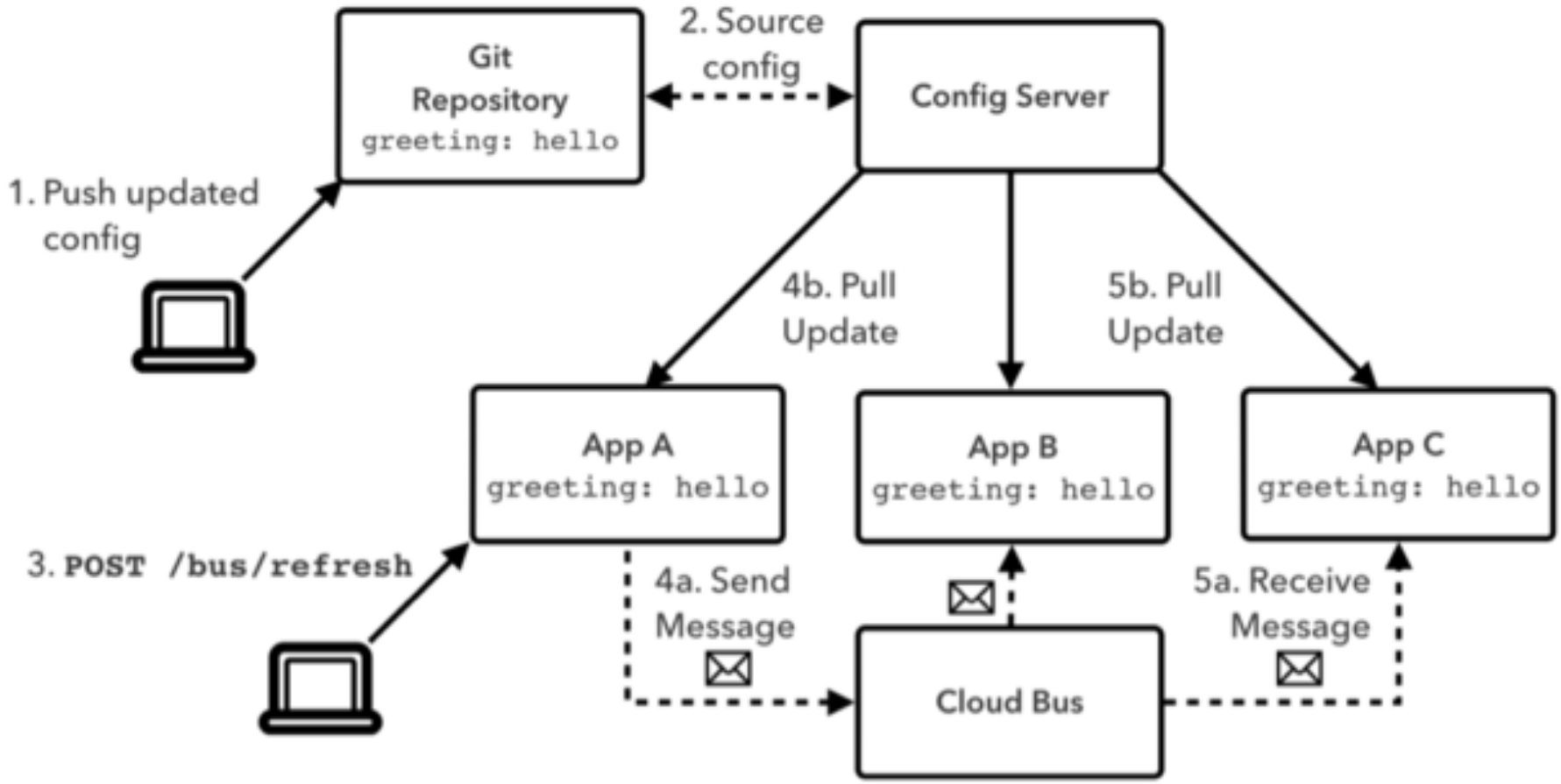
Spring Cloud Config

"**Gerenciamento de configuração** para microservices"

- Centraliza a configuração da aplicação
- Permite atualizações dinâmicas
- Versionado
- Suporte à rollback
- Suporta configuração via repositórios
  - *Git, SVN, filesystem*
- Permite atualização via barramento
  - ***Spring Cloud Bus***



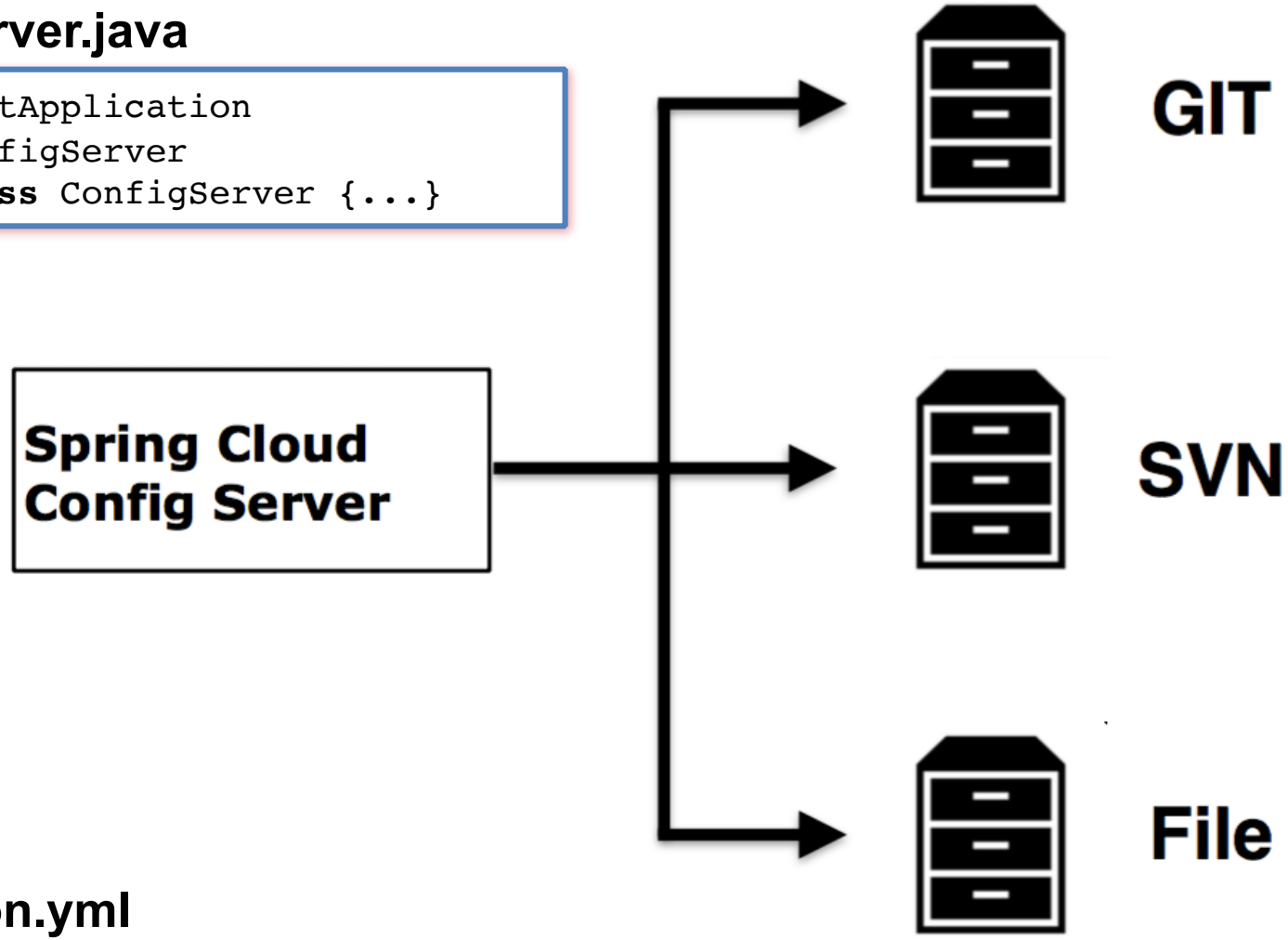
Spring Cloud Config



Spring Cloud Config **(Server)**

ConfigServer.java

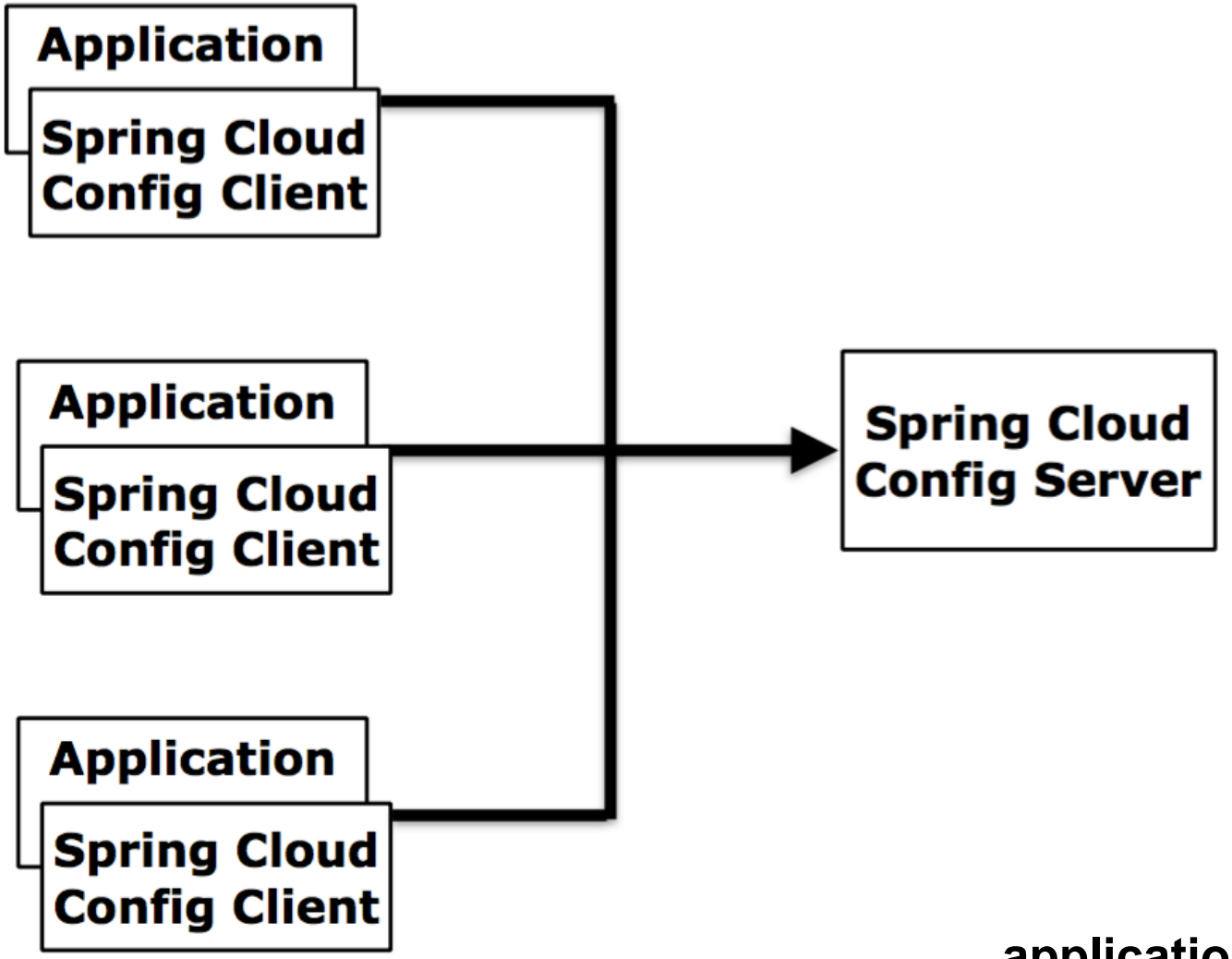
```
@SpringBootApplication
@EnableConfigServer
public class ConfigServer {...}
```



application.yml

```
spring.cloud.config.git.uri: https://github.com/...
```

Spring Cloud Config **(Client)**



application.yml

```
spring.cloud.config.uri: ${vcap.services.configserver.credentials.uri}
```

Demo

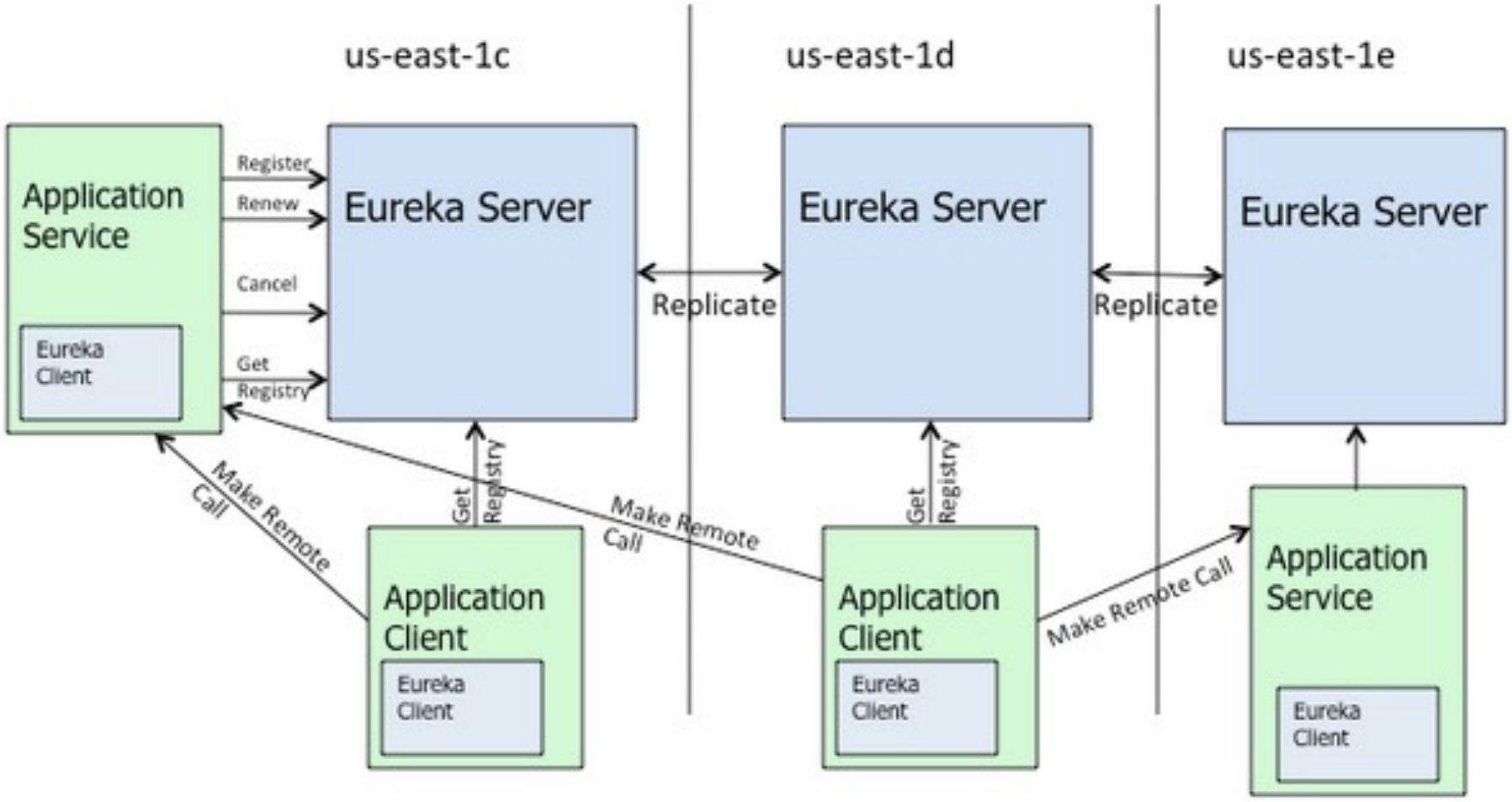
- **Gerenciamento de Configuração**
  - *Spring Cloud Config + Spring Cloud Bus*
  - *https://github.com/rcandidosilva/spring-cloud-sample*

Netflix Eureka

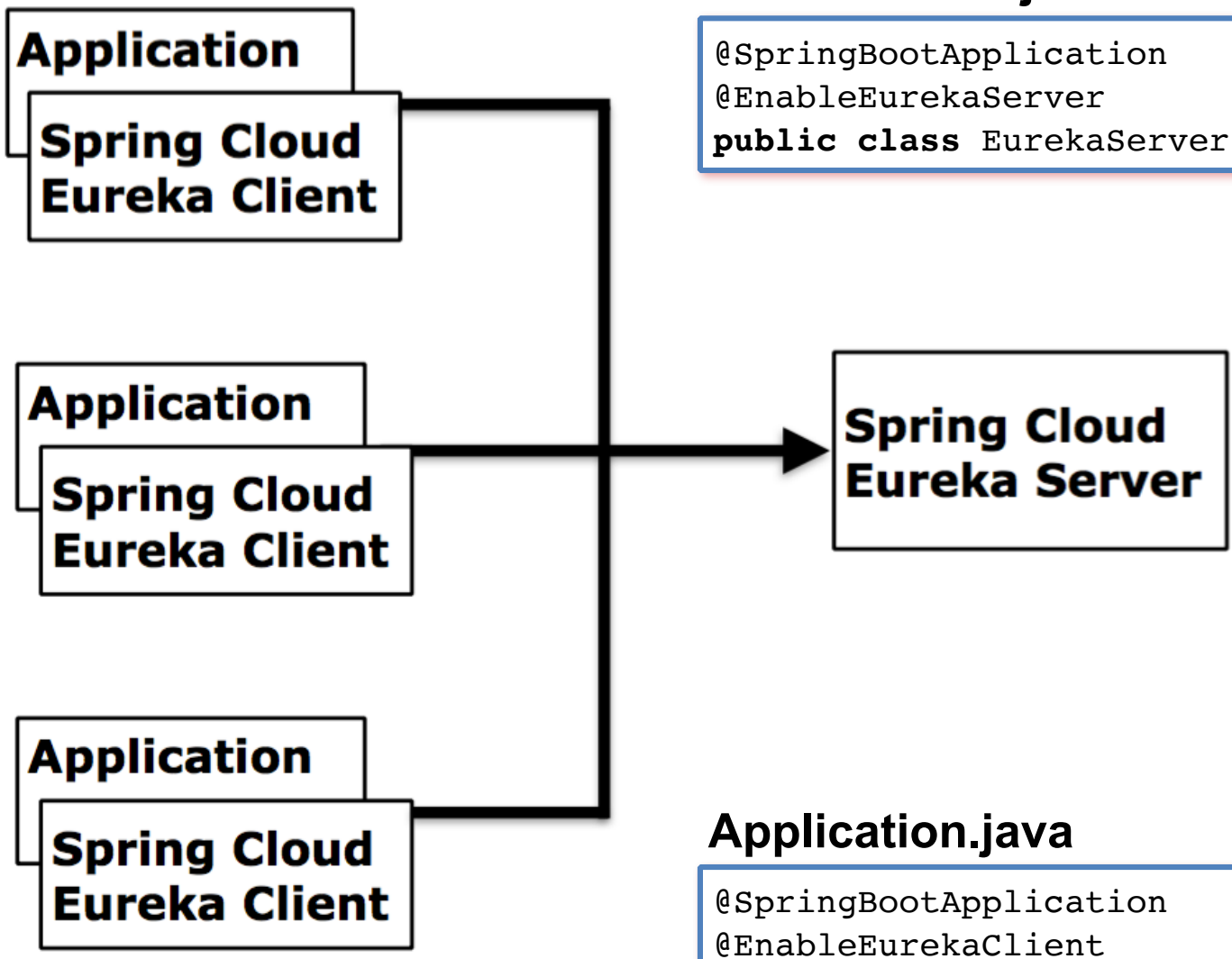
"**Transparência de localização** aos microservices"

- Registro de serviços ***REST based***
- Suporte à replicação
- Cache aplicado no *stub* cliente
- Resiliente
- Rápido… mas não consistente
- Fornece o alicerce para outros serviços
- Mantém registro de clientes com metadados

Netflix Eureka



Netflix Eureka



EurekaServer.java

```
@SpringBootApplication
@EnableEurekaServer
public class EurekaServer {...}
```

Application.java

```
@SpringBootApplication
@EnableEurekaClient
public class Application {...}
```

Demo

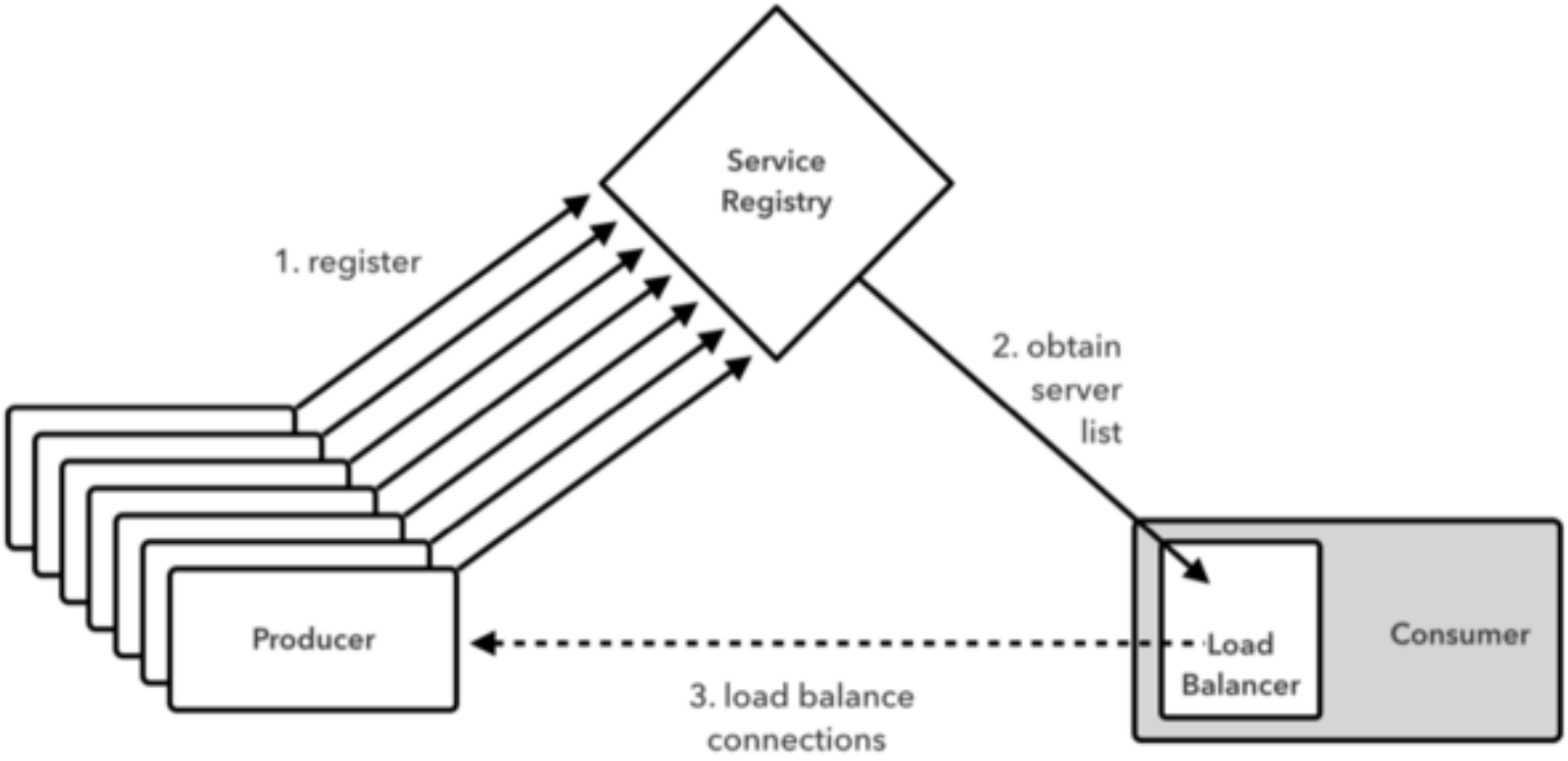
- **Transparência de Localização**
  - *Spring Cloud + Netflix Eureka*
  - *https://github.com/rcandidosilva/spring-cloud-sample*

Netflix Ribbon

"**Balanceamento de carga** para microservices"

- Balanceamento decentralizado no cliente
- Resiliente
- Suporte à tolerância a falhas
- Trabalha com múltiplos protocolos
  - *HTTP, TCP, UDP*
- Modelo assíncrono e reativo
- Suporte à caching e batching
- Múltiplos algoritmos de balanceamento

Netflix Ribbon



Demo

- **Balanceamento de Carga**
  - *Spring Cloud + Netflix Ribbon*
  - *https://github.com/rcandidosilva/spring-cloud-sample*

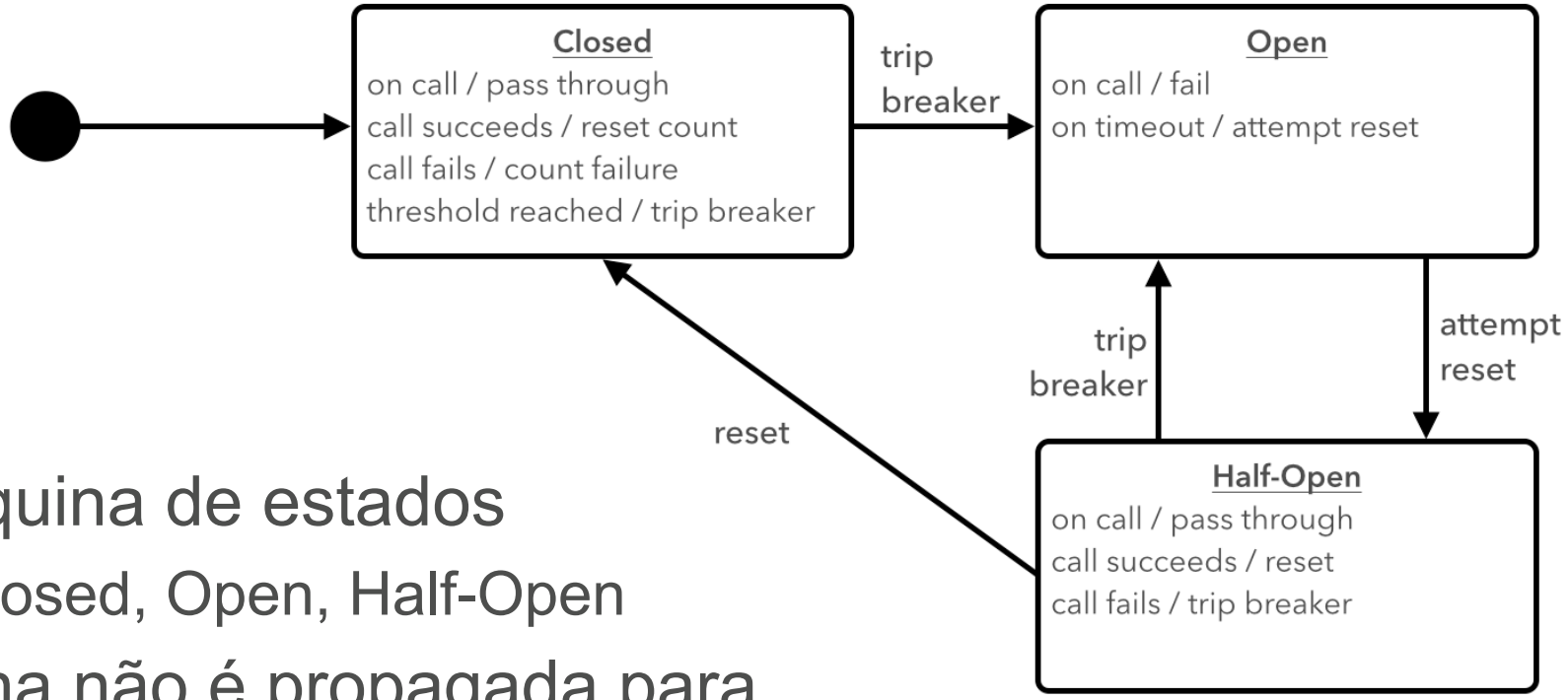
Netflix Hystrix

"**Tolerância à falhas** para microservices"

- Implementa padrão ***circuit breakers***
- Fornece monitoramento aos serviços
  - *Hystrix dashboard*
- Suporta comandos assíncronos
- Utiliza diferentes *thread pools*
- Pode implementar *timeouts*

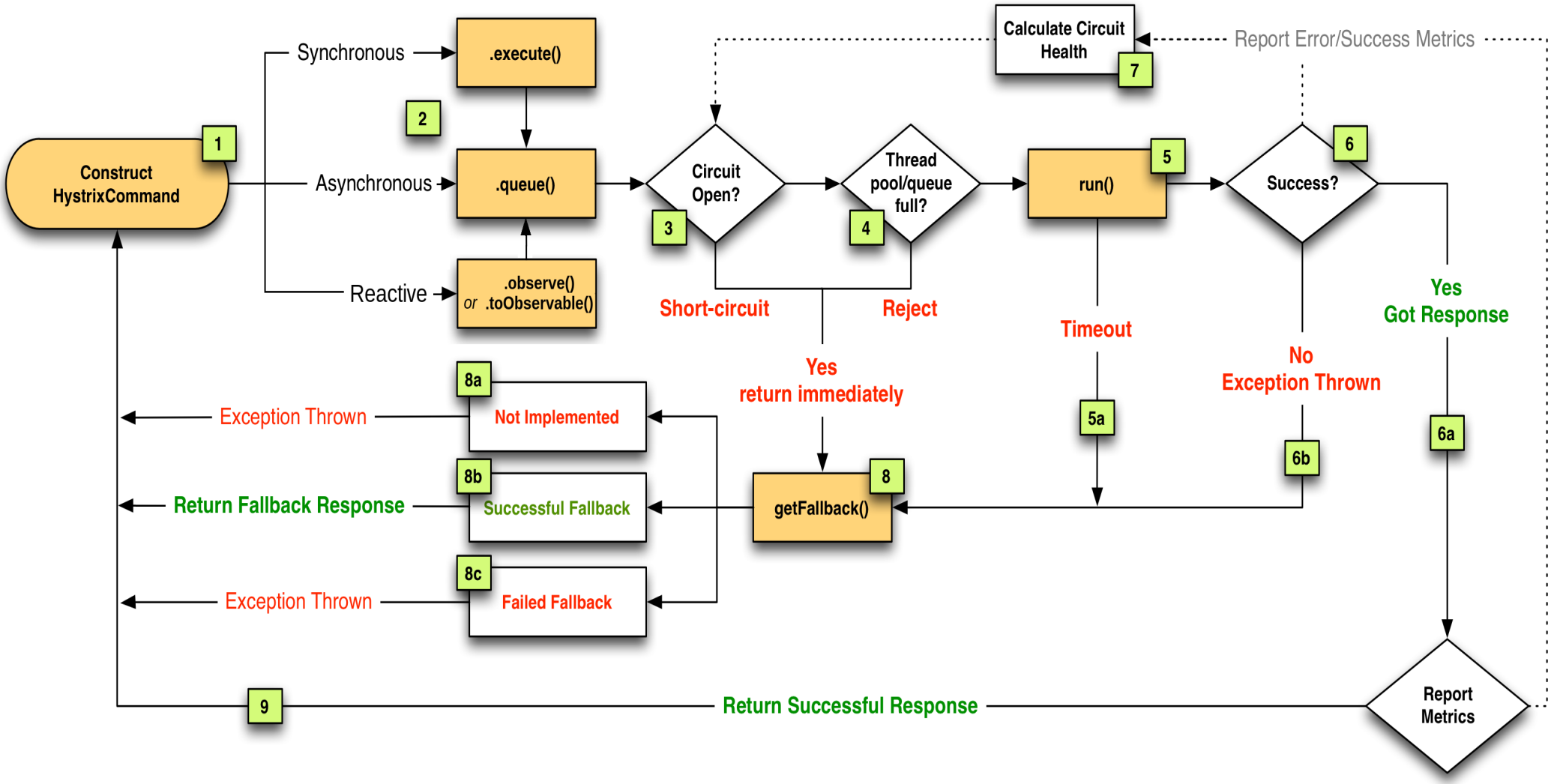
Netflix Hystrix

- **Circuit Breaker Pattern**

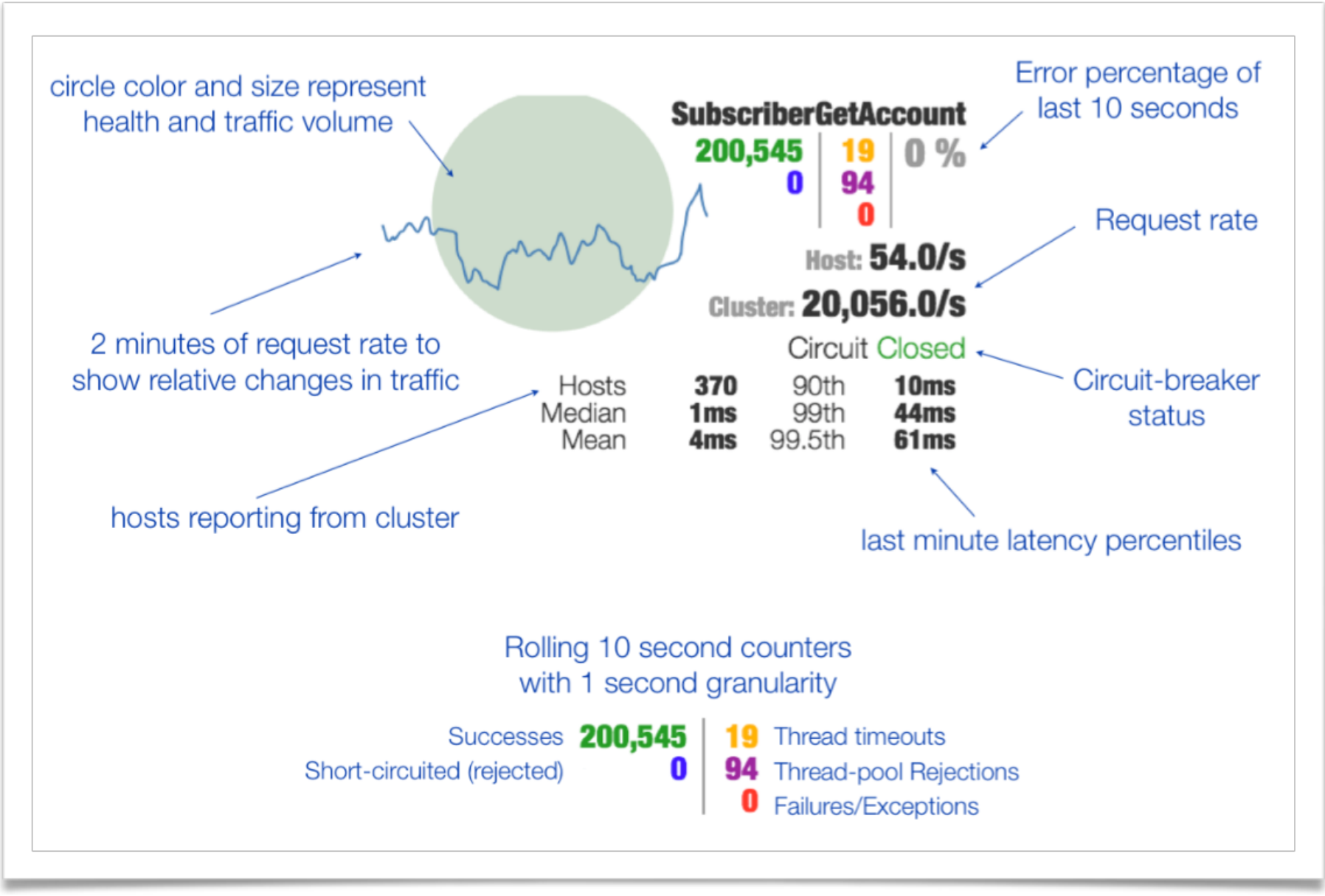


- Máquina de estados
  - Closed, Open, Half-Open
- Falha não é propagada para chamada do cliente

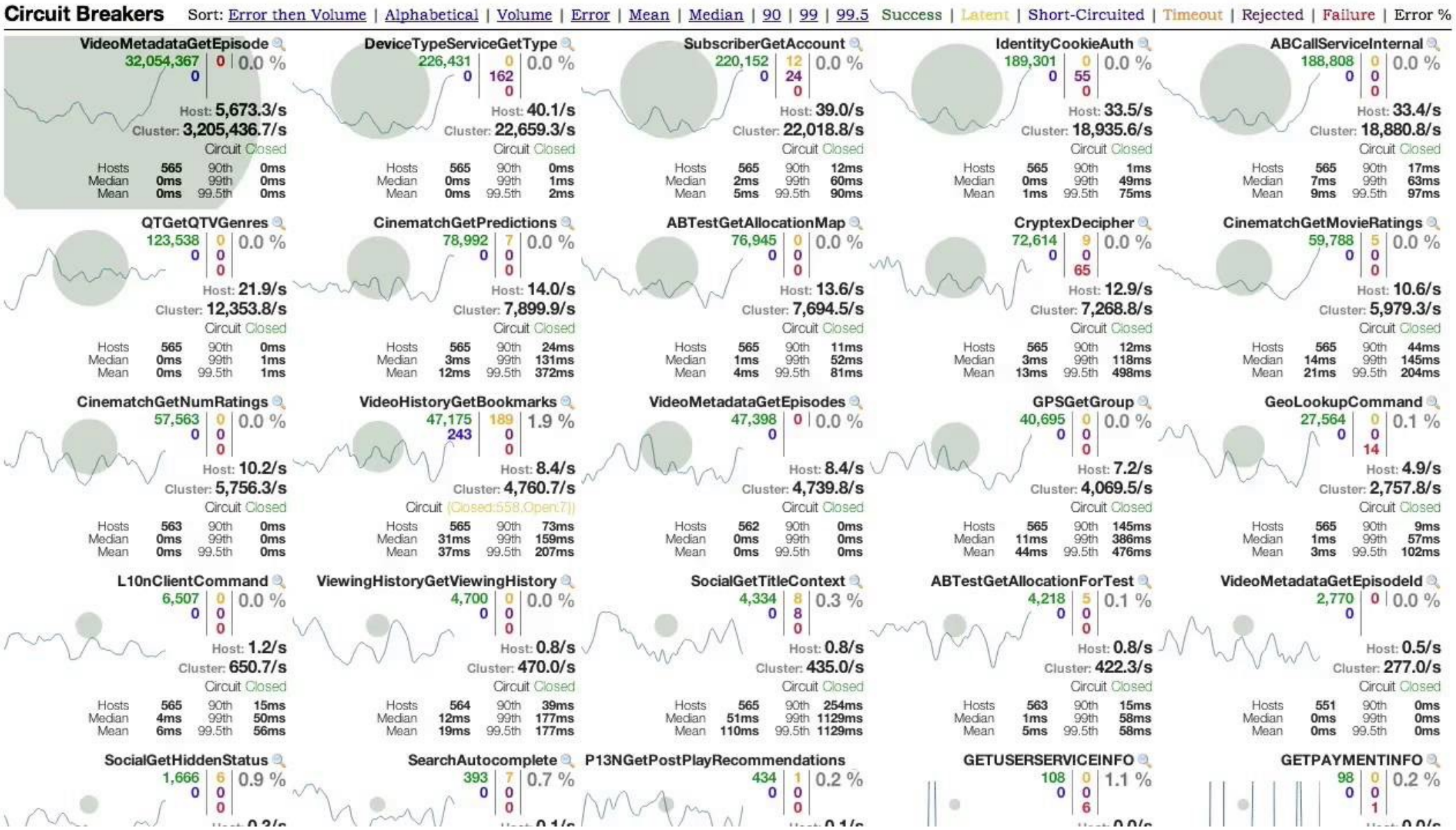
Netflix Hystrix



Hystrix Dashboard



Hystrix Dashboard + Turbine



Demo

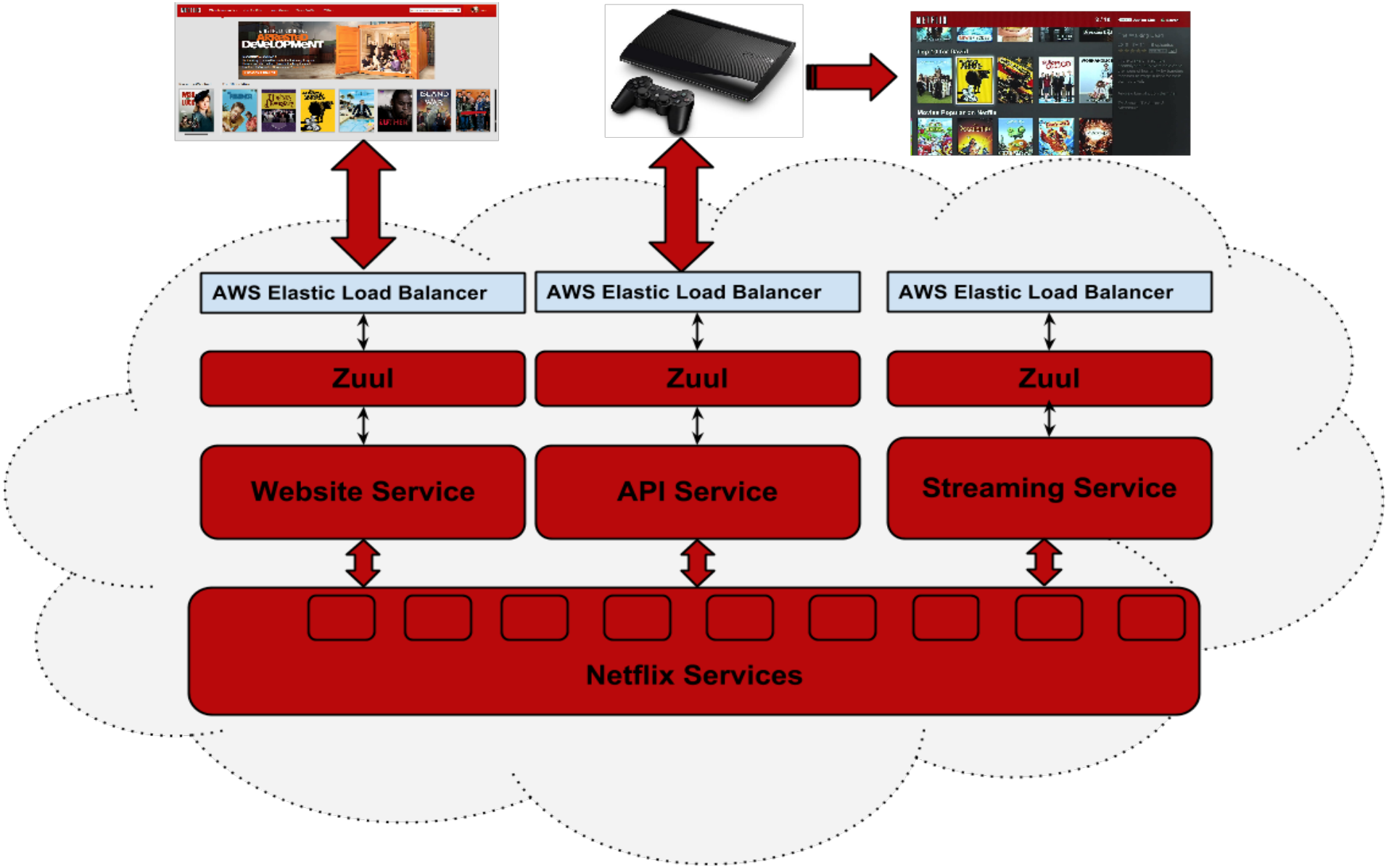
- **Tolerância a Falhas**
  - *Spring Cloud + Netflix Hystrix*
  - *https://github.com/rcandidosilva/spring-cloud-sample*

Netflix Zuul

"**Roteamento centralizado** para microservices"

- Fornece único ponto de entrada para os serviços
- Roteamento e balanceamento na JVM
- Cria uma rota para cada serviço no Eureka
- Define filtros para pontos de entrada
- Similar outros roteamentos
  - *httpd, nginx, CF go router*

Netflix Zuul



Demo

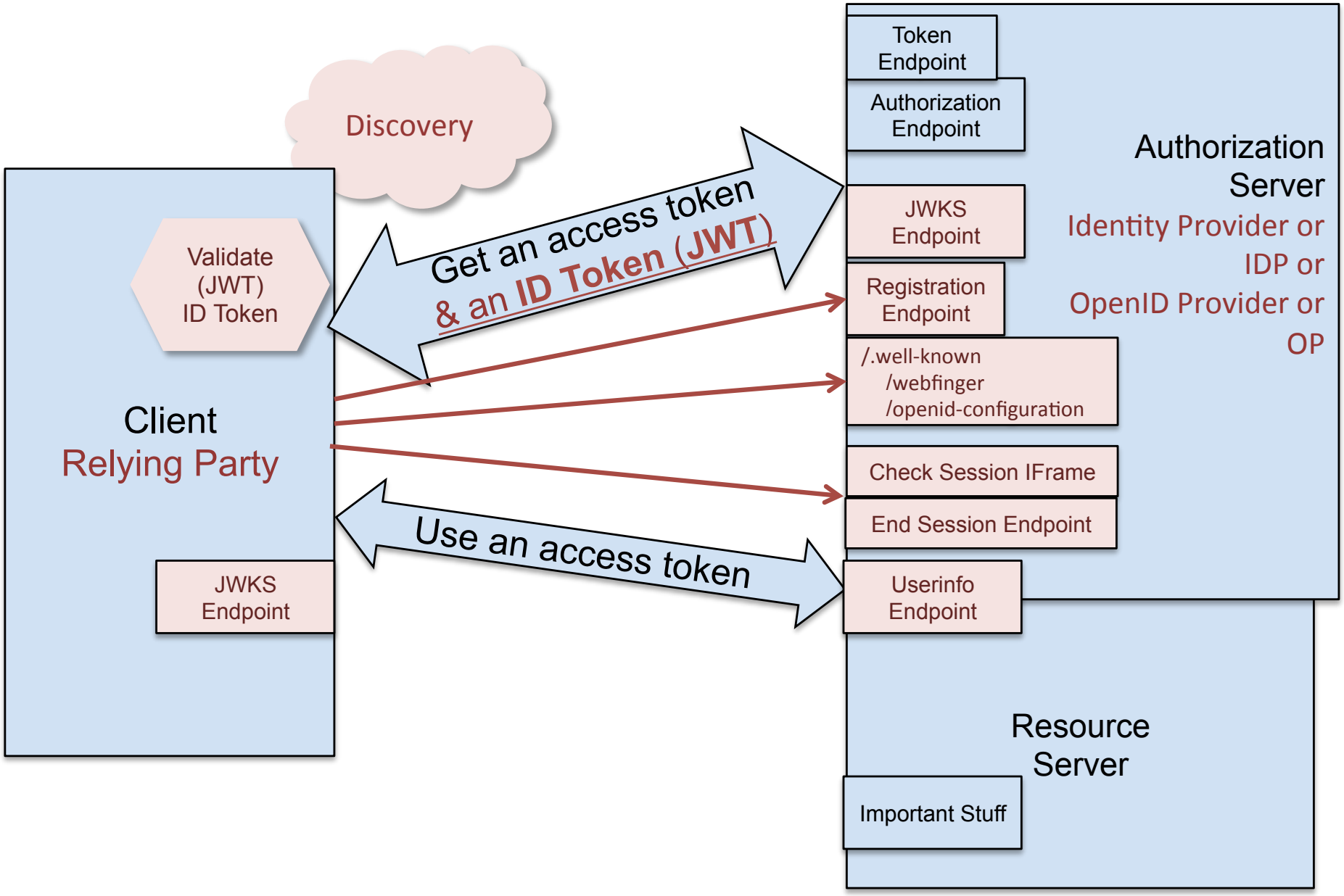
- **Roteamento centralizado**
  - *Spring Cloud + Netflix Zuul*
  - *https://github.com/rcandidosilva/spring-cloud-sample*

Spring Cloud Security

"**Segurança aplicada** para microservices"

- Integração Spring Security + OAuth2
- SSO com OAuth2 e OpenID Connect
- Proteção dos serviços com tokens (JWT)
- Transmissão tokens entre SSO e apps
- **OAuth2 + OpenID Connect + JWT ;)**

Spring Cloud Security



Conclusões…

- Microservices são sistemas distribuídos
- Sistemas distribuídos são complexos
- Netflix OSS define ótimas ferramentas para implementação com microservices
- Spring Cloud
  - Ótima abstração para Netflix OSS
  - Fácil utilização (via anotações)
  - Integração com ecossistema Spring
  - **Enjoy it ;)**

Perguntas

Referências

- http://projects.spring.io/spring-boot/
- http://projects.spring.io/spring-cloud/
- https://netflix.github.io/
- https://github.com/Netflix/zuul
- https://github.com/Netflix/eureka
- https://github.com/Netflix/ribbon
- https://github.com/Netflix/Hystrix
- http://www.pwc.com/us/en/technology-forecast/2014/cloud-computing/features/microservices.html
- http://martinfowler.com/articles/microservices.html
- http://callistaenterprise.se/blogg/teknik/2015/04/10/building-microservices-with-spring-cloud-and-netflix-oss-part-1/
- http://www.javaworld.com/article/2927920/cloud-computing/build-self-healing-distributed-systems-with-spring-cloud.html

Muito obrigado!

@rcandidosilva

rodrigocandido.me