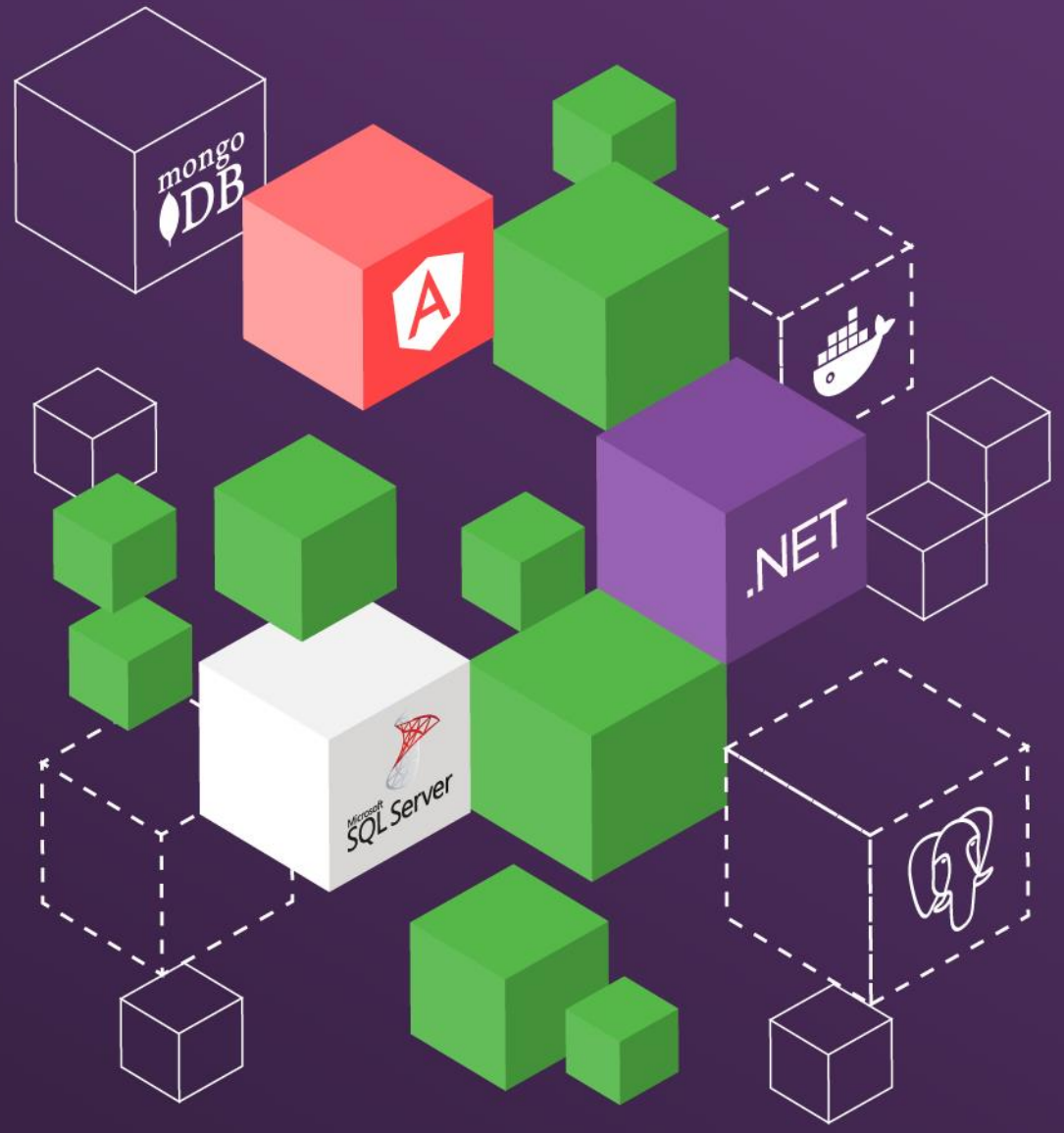


BIENVENIDOS
AL CURSO:

Arquitectura de Microservicios en Net

SESIÓN 06

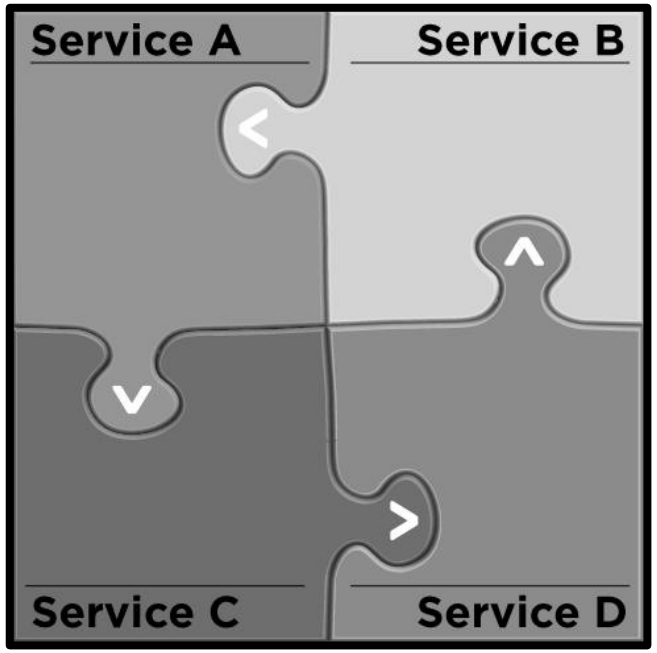
GALAXY
TRAINING

ÍNDICE

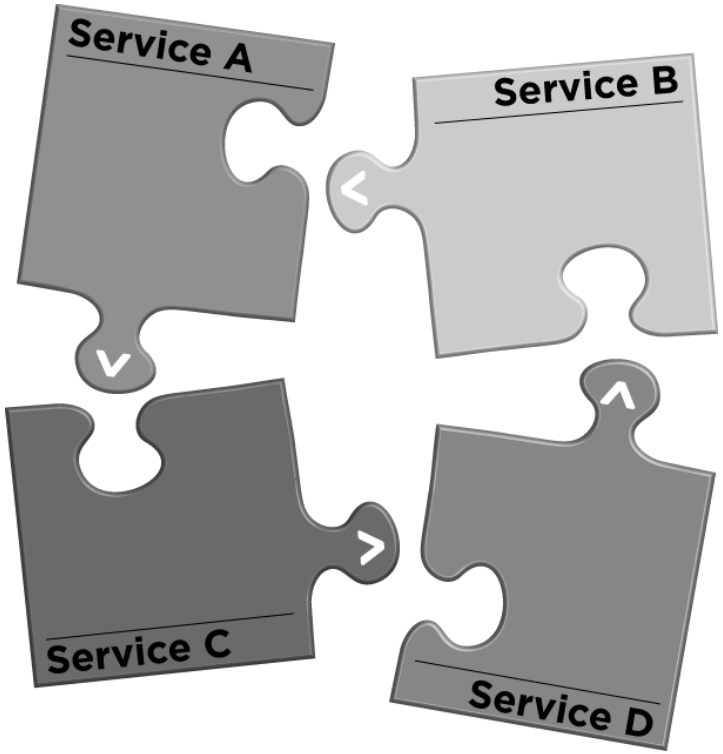
Encontrar servicios utilizando el Services Discovery

¿Qué es Service Discovery y por qué lo necesitamos?

Cambios en la forma en que desarrollamos software



Desde una aplicación monolítica

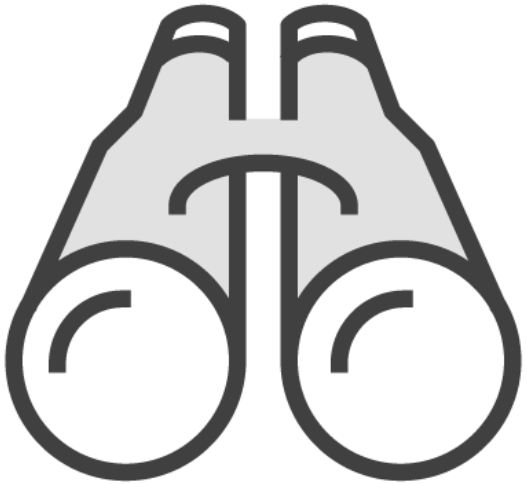


Para servicios desplegables individualmente

Registro y Discovery de microservicios.

El problema: ¿cómo un servicio ubica a otro?

Servicio de aplicación A



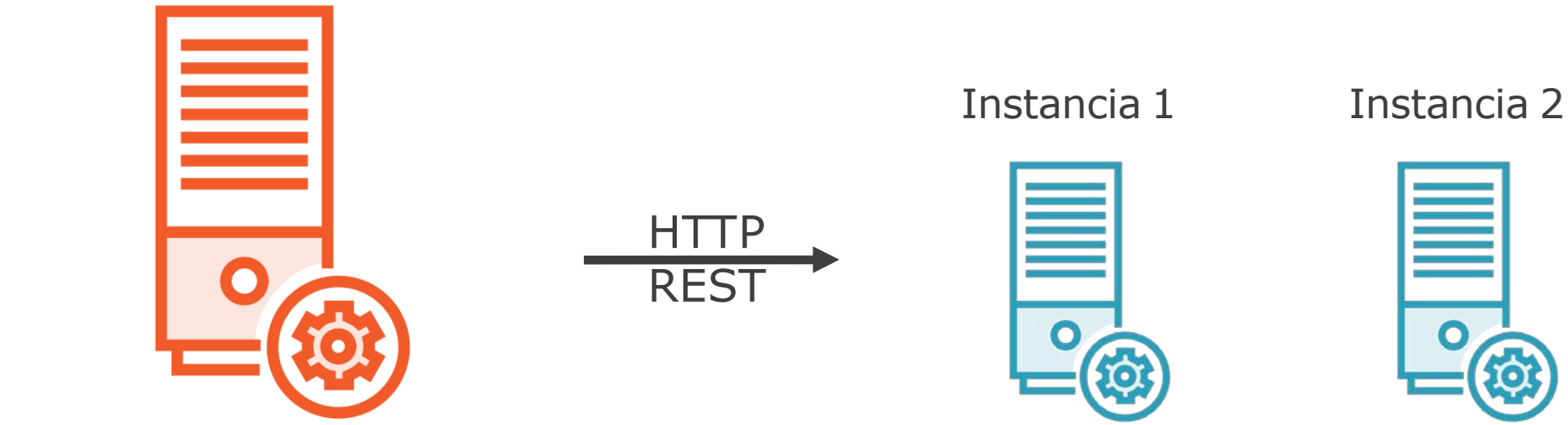
Localización?

Servicio de aplicación B

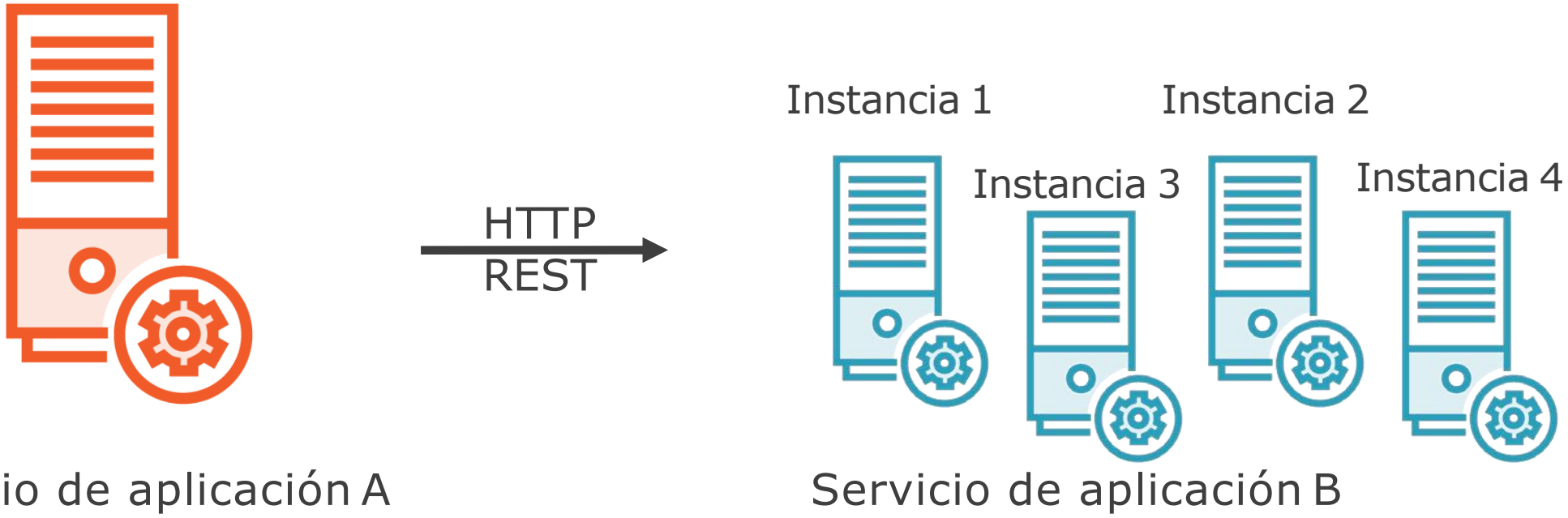
Registro y Discovery de microservicios.

El enfoque simple: a través de la configuración

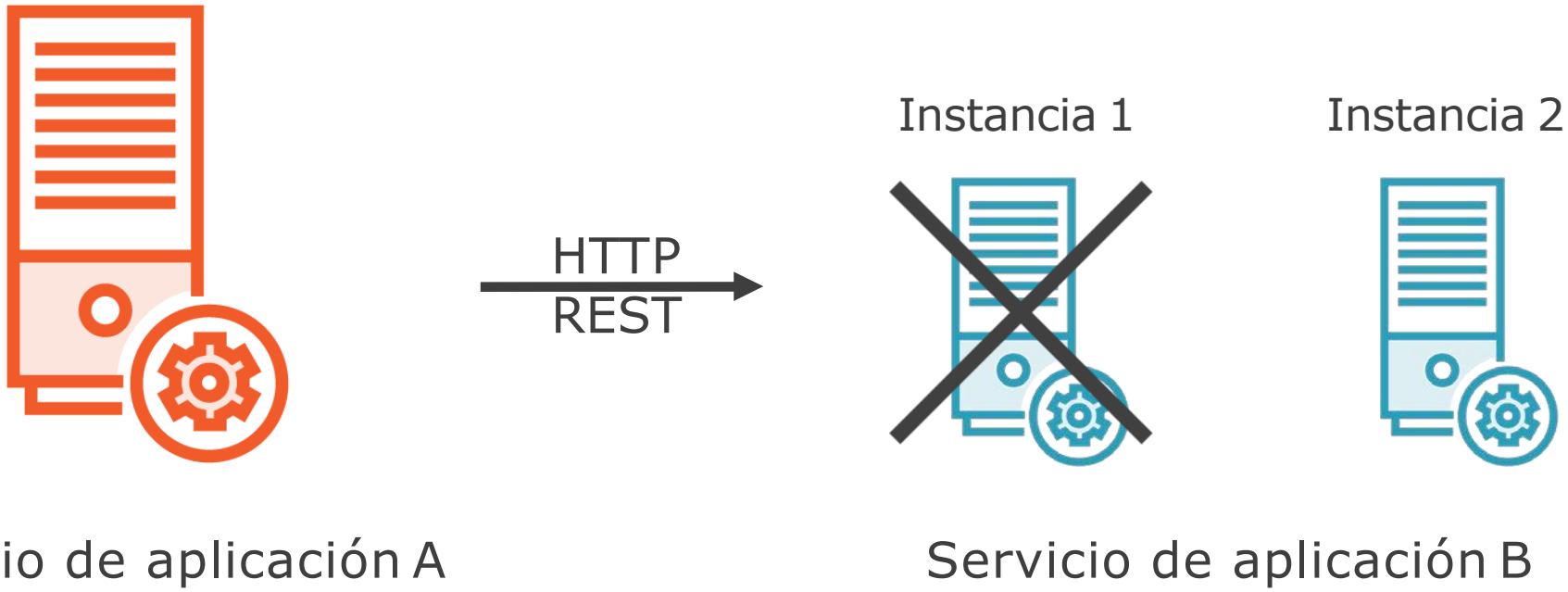
Multiple Instancias



Las instancias van y vienen en respuesta a la demanda



Las instancias fallan



El enfoque simple: a través de la configuración

¡El enfoque simple es **demasiado estático** (congelado en el tiempo) para la nube!

El enfoque simple: a través de la configuración

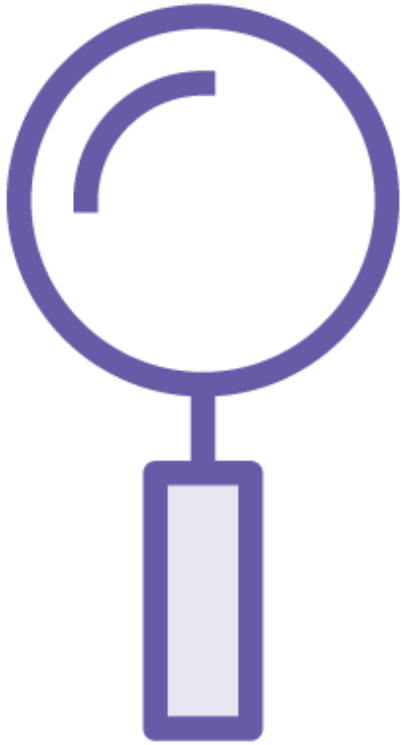
El descubrimiento de servicios proporciona

- Una forma para que un servicio se registre
- Una forma para que un servicio se desregistre
- Una manera para que un cliente encuentre otros servicios
- Una forma de verificar el estado de un servicio y eliminar instancias caidas

Discovering Services con Spring Cloud

Registro y Discovery de microservicios.

El enfoque simple: a través de la configuración



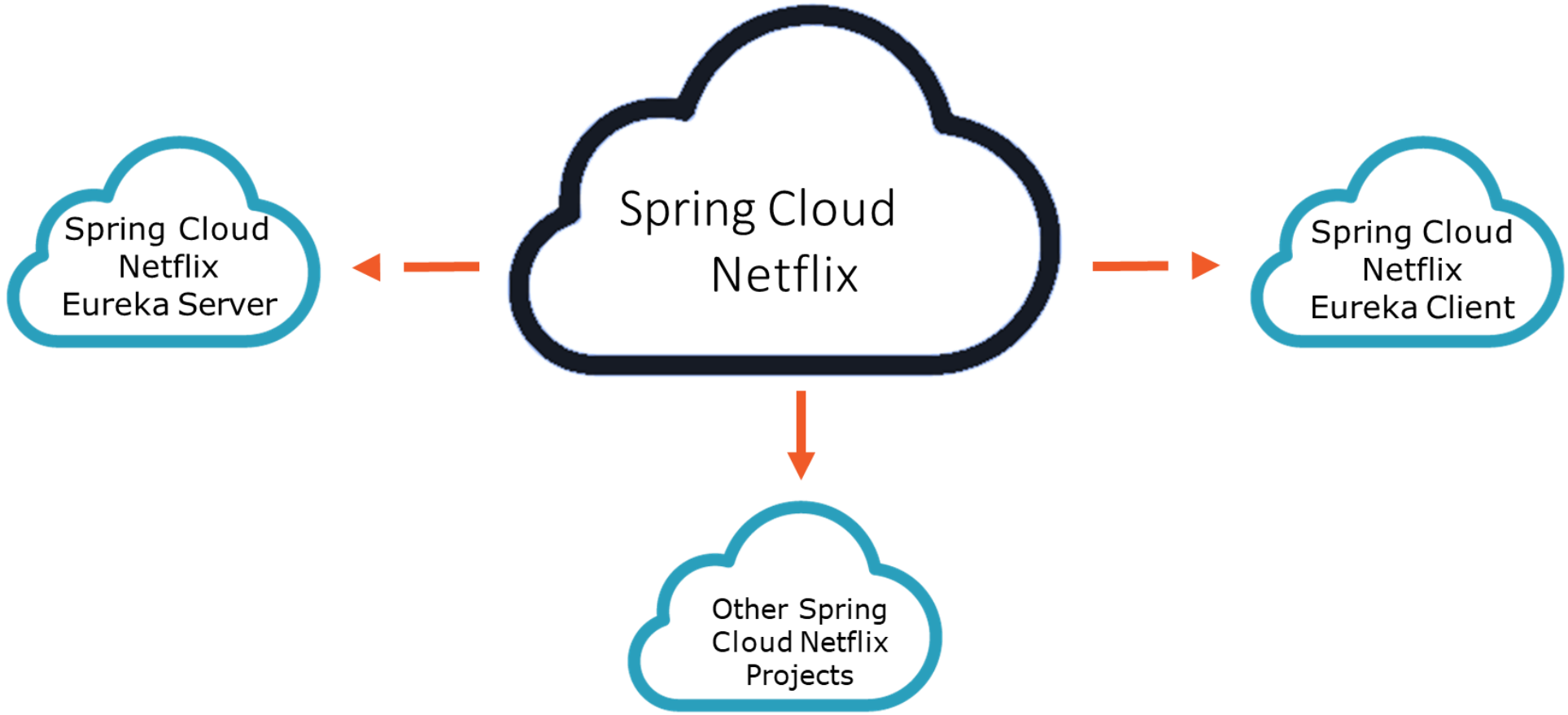
Discover services con:

- Spring Cloud Consul
- Spring Cloud Zookeeper
- Spring Cloud Netflix

Netflix OSS + Spring + Spring Boot

=

Spring Cloud Netflix



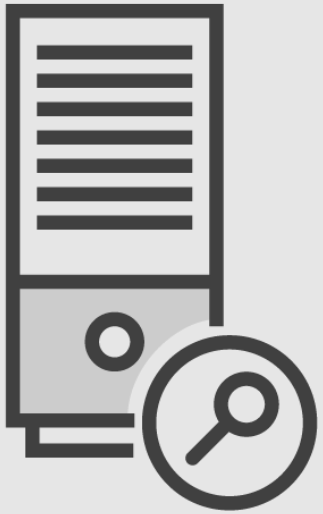
Registro y Discovery de microservicios.

Componentes clave en Service Discovery

Discovery Server

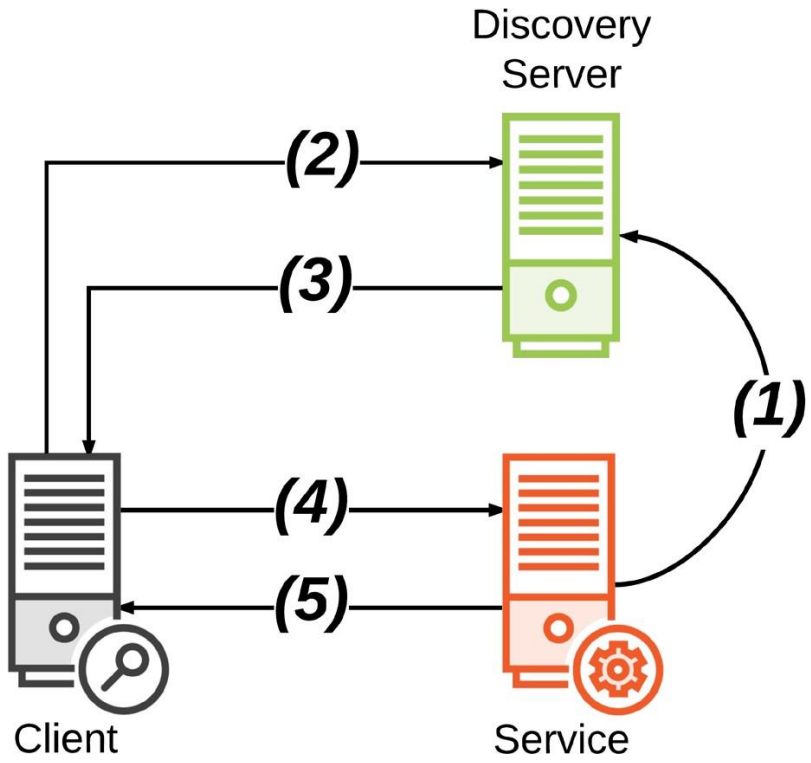
Service

Client



El enfoque simple: a través de la configuración

1. El servicio registra la ubicación
2. El cliente busca la ubicación del servicio
3. El servidor de descubrimiento devuelve la ubicación
4. El cliente solicita servicio en la ubicación
5. El servicio envía respuesta



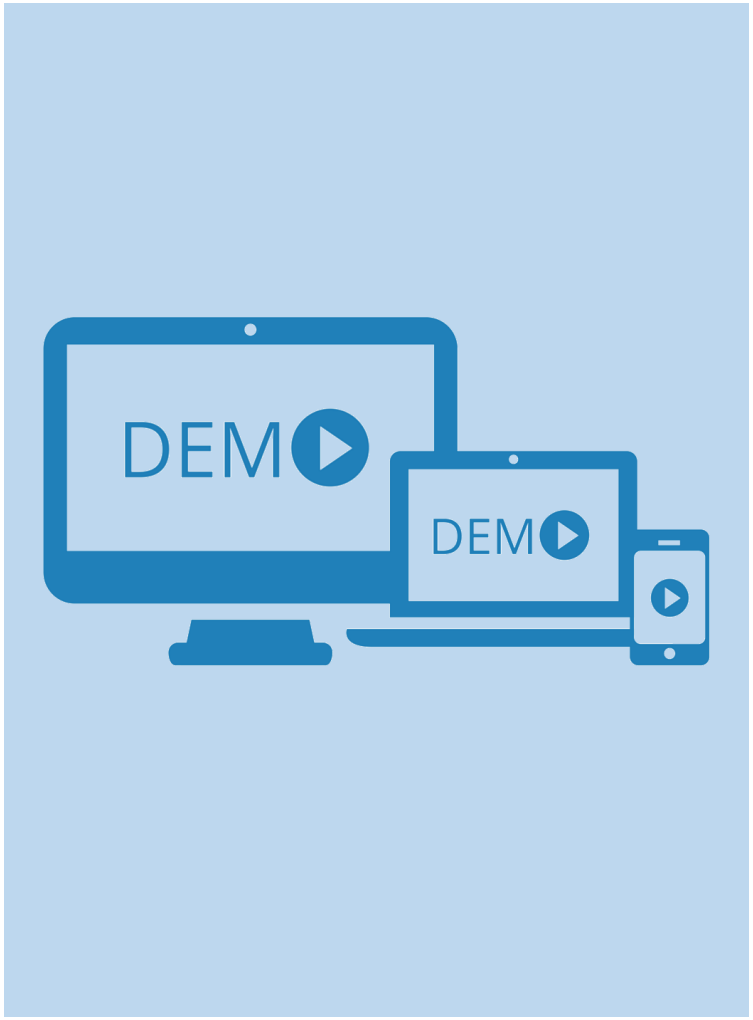
El enfoque simple: a través de la configuración

Un registro de ubicaciones de servicio gestionado activamente

Un solo origen para una o más instancias

Proyecto Spring Cloud:

- Servidor Spring Cloud Eureka



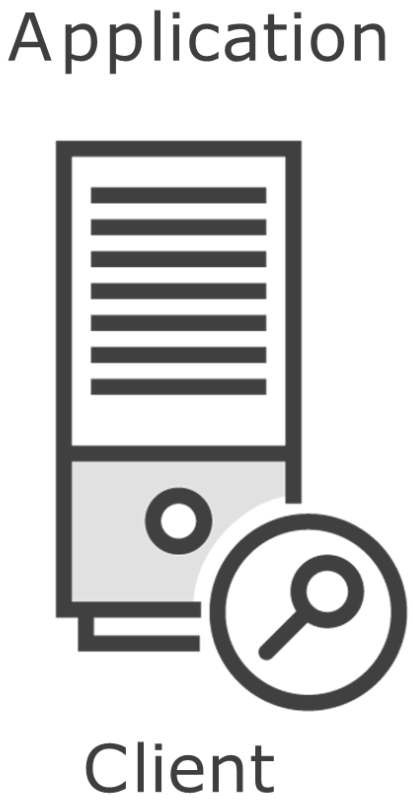
Creando e iniciando un discovery server

Servicio de que se conecta a un Discovery Server

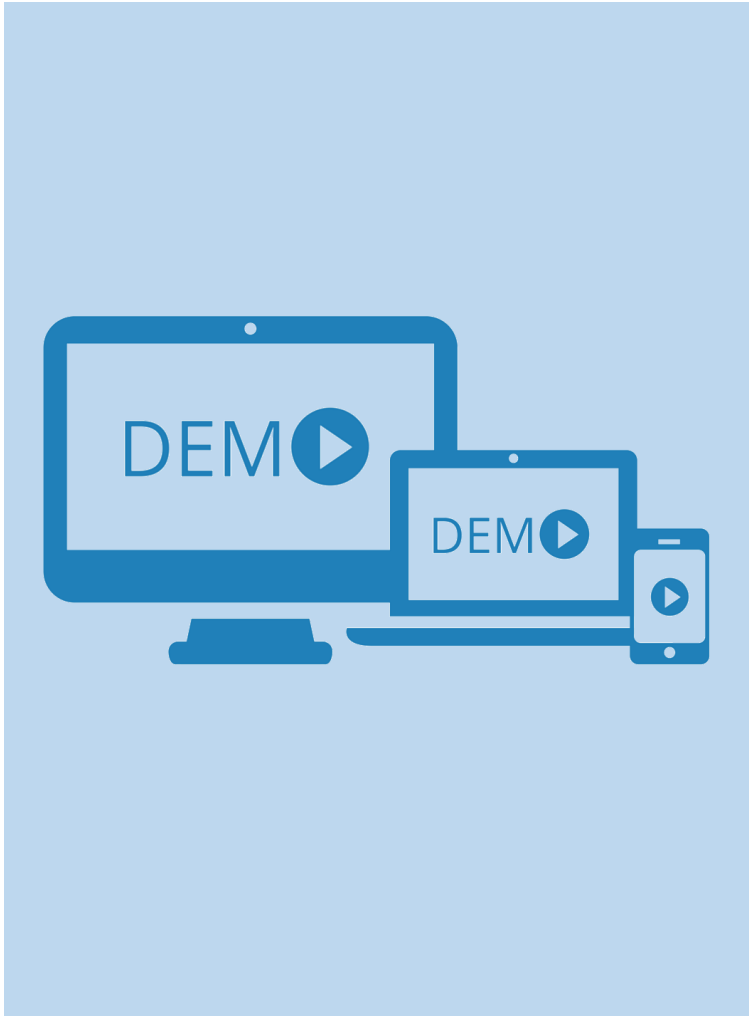
- Proporcionar algunas funciones de la aplicación.
- El receptor de request
- Una dependencia de otros servicios Una o más instancias
- Uso del discovery client
  - Registrarse
  - Darse de baja

Creando e iniciando un discovery server

Cliente que consume a un Discovery Server



- Llama a otro servicio de aplicación para implementar su funcionalidad
- El emisor de las solicitudes depende de otros servicios Usuario del cliente de descubrimiento
- Encuentra ubicaciones de servicio

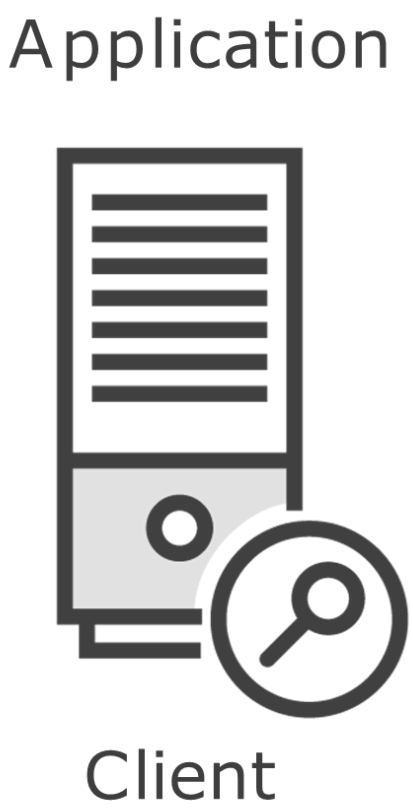


Creating a client that can discover services

Spring Cloud Eureka Dashboard

Registro y Discovery de microservicios.

Cliente que consume a un Discovery Server



Habilitado por defecto

- eureka.dashboard.enabled=true

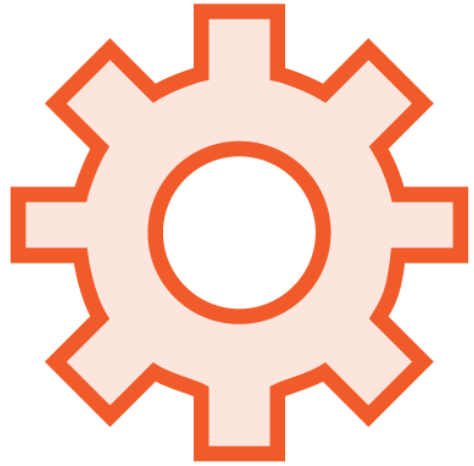
Muestra metadatos útiles y estado del servicio

Servidor Eureka: ¿Estan saludables mis servicios?

Comprueba regularmente el estado de los servicios.

Los clientes envían latidos cada 30 segundos (predeterminado)

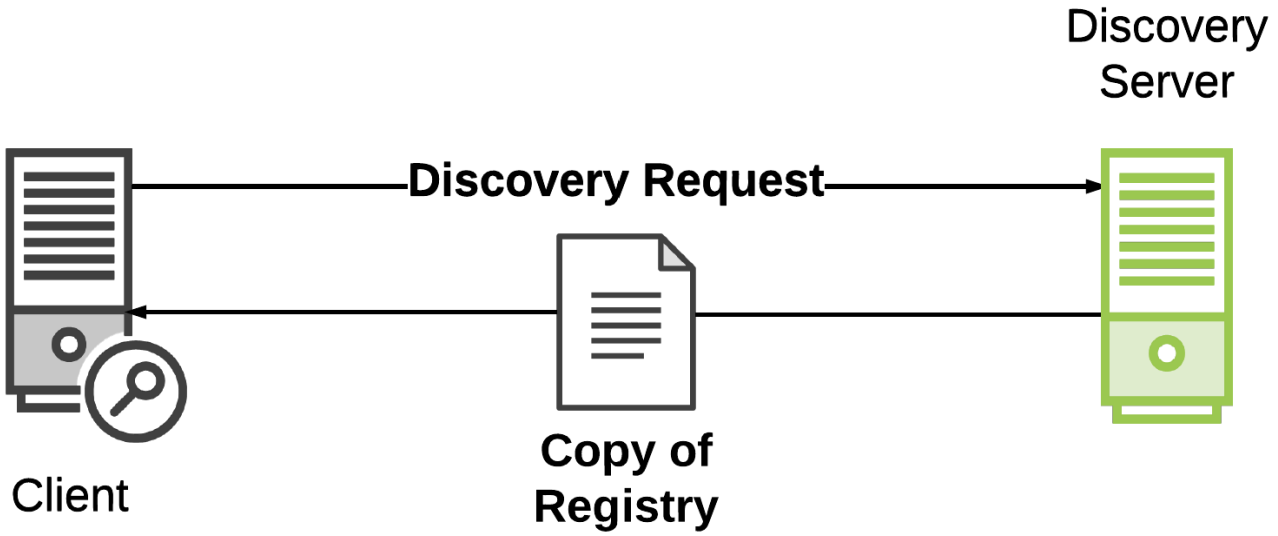
Servicios eliminados después de 90 segundos sin latidos (predeterminado)



Puede personalizar la configuración para usar /health endpoint eureka.client. healthcheck.enabled

Servidor Eureka: ¿Estan saludables mis servicios?

- El registro se distribuye (almacena en caché localmente en cada cliente)
- Los clientes pueden operar sin servidor de descubrimiento
- Obtiene deltas para actualizar el registro



Spring Cloud Eureka AWS Support

Registro y Discovery de microservicios.

Spring Cloud Eureka AWS Support

AWS-specific instance data

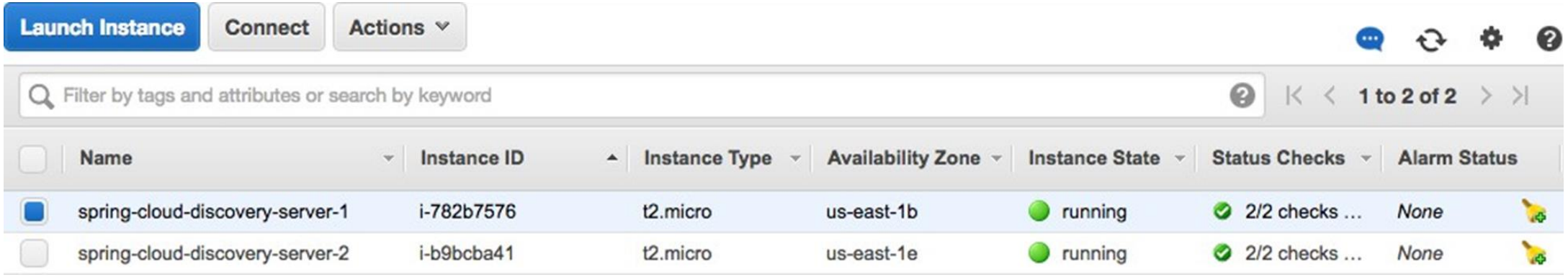
Multi-zone aware

Multi-region aware

Elastic IP Binding

EC2 Dashboard

us-east-1



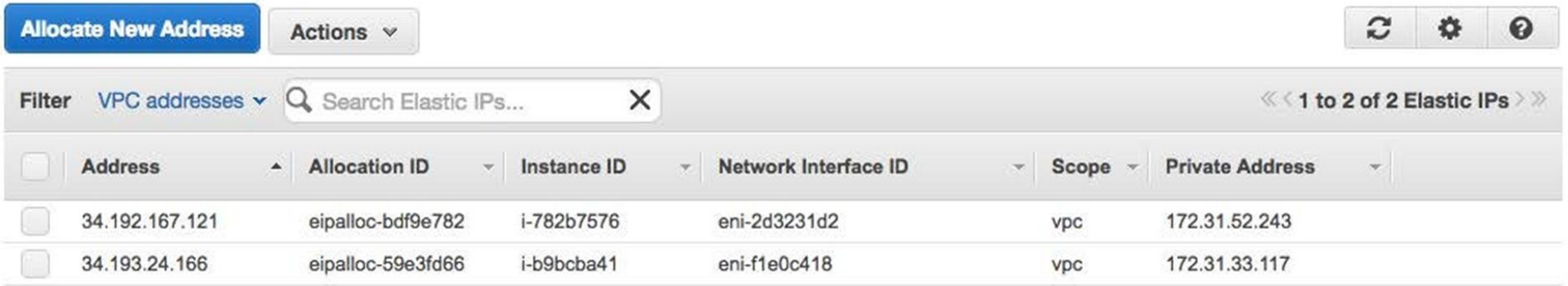

```
eureka.client.availability-zones.us-east-1=us-east-1b,us-east-1e
```

Availability Zones Configuration in `application.properties`

```
eureka.client.availability-zones.[region]=[az1],[az2],[az3]
```

Elastic IP Dashboard

us-east-1




```
eureka.client.service-url.us-east-1b=
                        http://ec2-34-192-167-121.compute-1.amazonaws.com:8761/eureka/
eureka.client.service-url.us-east-1e=
                        http://ec2-34-193-24-166.compute-1.amazonaws.com:8761/eureka/
```

Service URL Configuration in `application.properties`

`eureka.client.service-url.[zone]=http://[eip-dns]/eureka`

- *Use EIP DNS name. Do not use IP (as of version Eureka 1.4)

Eureka Dashboard: AWS Multi-zone Discovery Servers

DS Replicas

ec2-34-192-167-121.compute-1.amazonaws.com

ec2-34-193-24-166.compute-1.amazonaws.com

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
DISCOVERY-SERVER-1	ami-40d28157 (1)	us-east-1b (1)	UP (1) - i-782b7576
DISCOVERY-SERVER-2	ami-40d28157 (1)	us-east-1e (1)	UP (1) - i-b9bcba41

Eureka Dashboard: AWS Instance Data

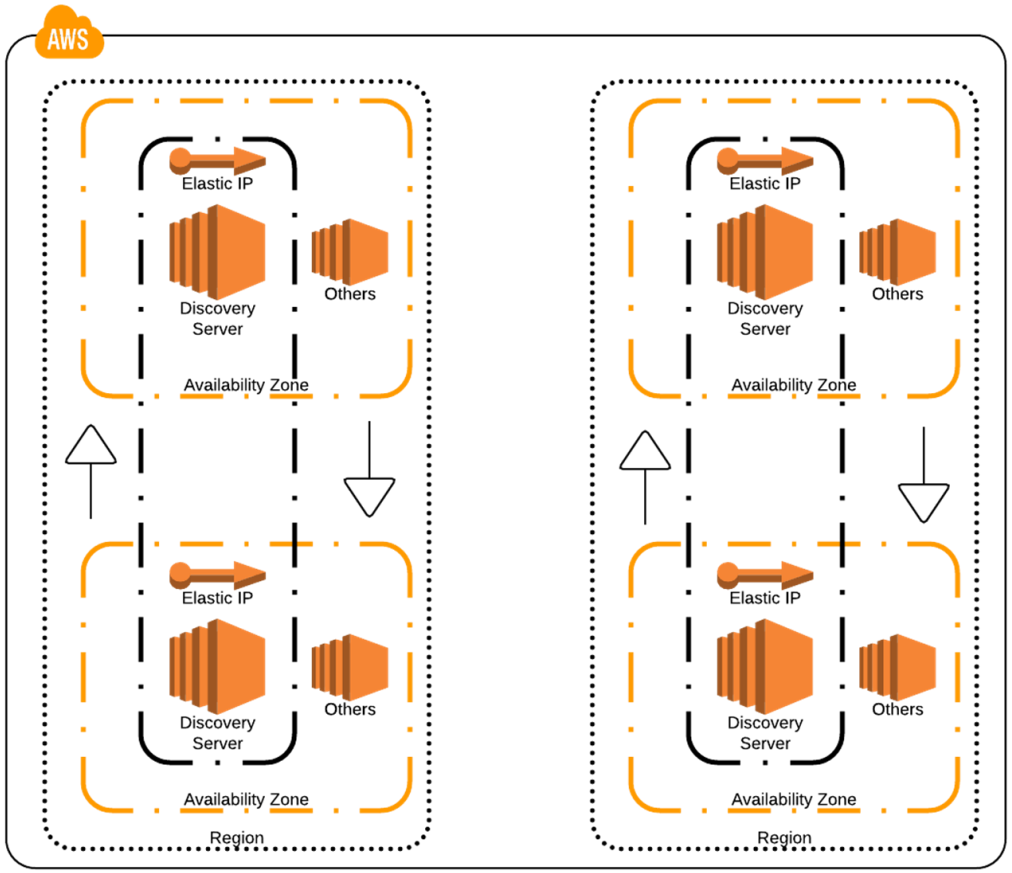
Instance Info

Name	Value
public-ipv4	34.192.167.121
public-hostname	ec2-34-192-167-121.compute-1.amazonaws.com
instance-id	i-782b7576
instance-type	t2.micro
ami-id	ami-40d28157
ipAddr	172.31.52.243
status	UP
availability-zone	us-east-1b

Eureka Dashboard: AWS Instance Data

One or more discovery servers (DS) per zone

One EIP (static) per Discovery Server



Auto scaling groups evenly across zones

GRACIAS

POR SU PREFERENCIA

GALAXY
TRAINING