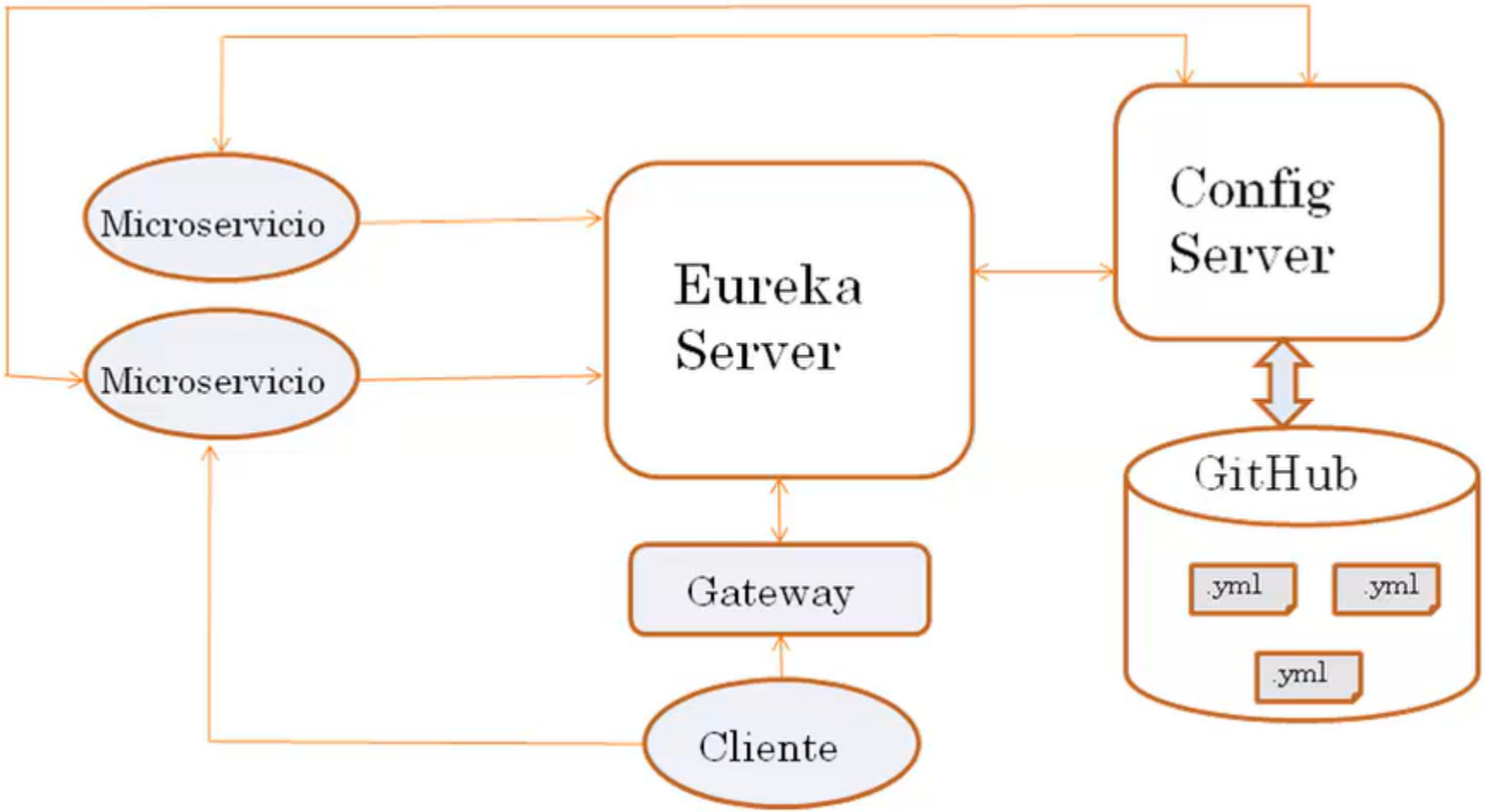
Spring Cloud

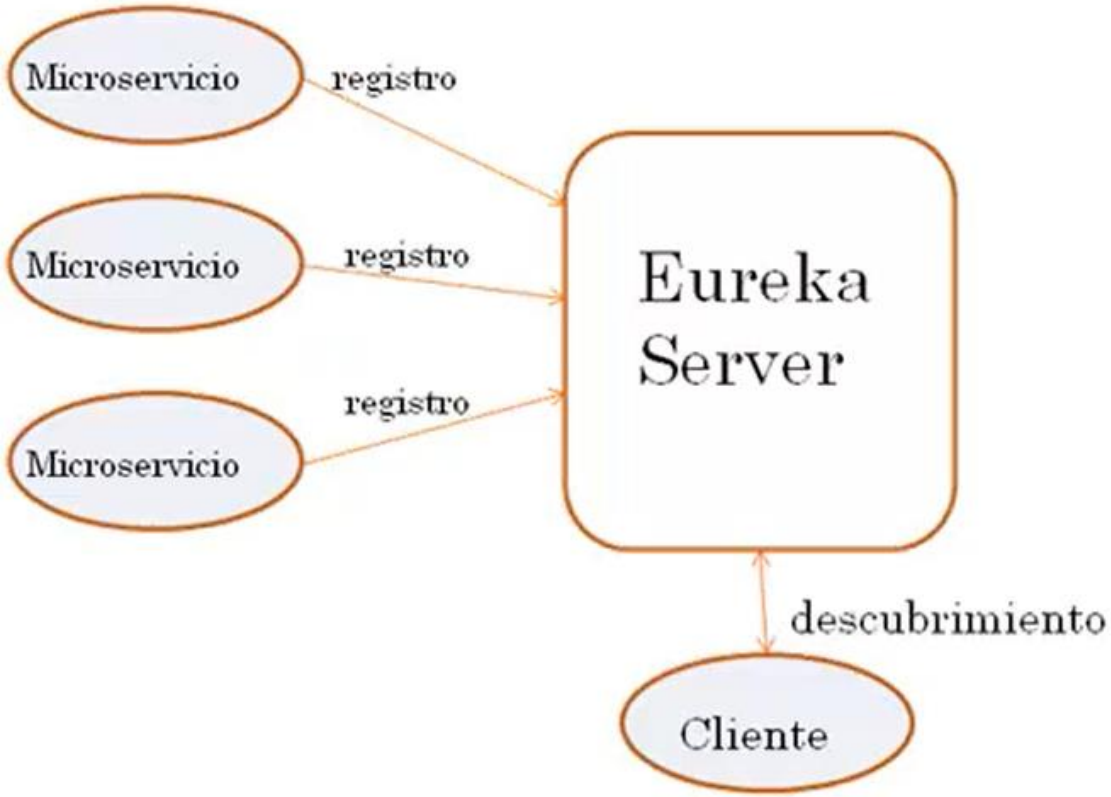
Tecnologías

- **Eureka server**: Servicio en el que se registran microservicios para facilitar el descubrimiento y utilización de los mismos. Los clientes no necesitan conocer la dirección
- **Ribbon**: Librería del lado cliente que proporciona balanceo de carga en el acceso a servicios. Permite balanceo de carga, mejora de rendimiento.
- **Spring Cloud Config**: servidor que centraliza la configuración de un conjunto de servicios en un único lugar. Facilita el mantenimiento.
- **Gateway**: Punto de acceso único a un conjunto de servicios desde las aplicaciones clientes finales.


Microservicio
Microservicio
Eureka Server
Config Server
GitHub
.yml
.yml
.yml
Gateway
Cliente

Eureka server

- Servidor para registro y descubrimiento de servicios.
- Permite acceder a los servicios sin conocer su ubicación real y puerto.



Eureka server

- Se crea como microservicio Spring Boot
- Requiere la dependencia **Eureka server:**

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
```

- Clase main anotada con **@EnableEurekaServer** (clase lanzadora)

```
@EnableEurekaServer
@SpringBootApplication
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Eureka server

- Configuraciones del servidor eureka en **application.properties**

```
spring.application.name=eureka-server
server.port=8761
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
```

Eureka server

- Si el fichero es **application.yml** el formato es:

```
eureka:
  client:
    fetch-registry: false
    register-with-eureka: false
server:
  port: 8761
spring:
  application:
    name: eureka-server
```

Registro de servicios en Eureka

- Requiere dependencia **Eureka Discovery Client**

```
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
```

(proyecto>botón derecho>Spring >Add Starters)

y elegir **Eureka Discovery Client**)

- Por defecto todos los servicios que se conecten con Eureka se van a registrar automaticamente.

Registro de servicios en Eureka

- **En el archivo de configuración (application.properties o application.yml)** se indica el nombre de registro del servicio y la url del servidor Eureka:
- En **application.properties** habrá que añadir:

```
spring.application.name=identificadorespecificodentrodeeureka
eureka.client.service-url.defaultZone=http://localhost:8761/eureka
```

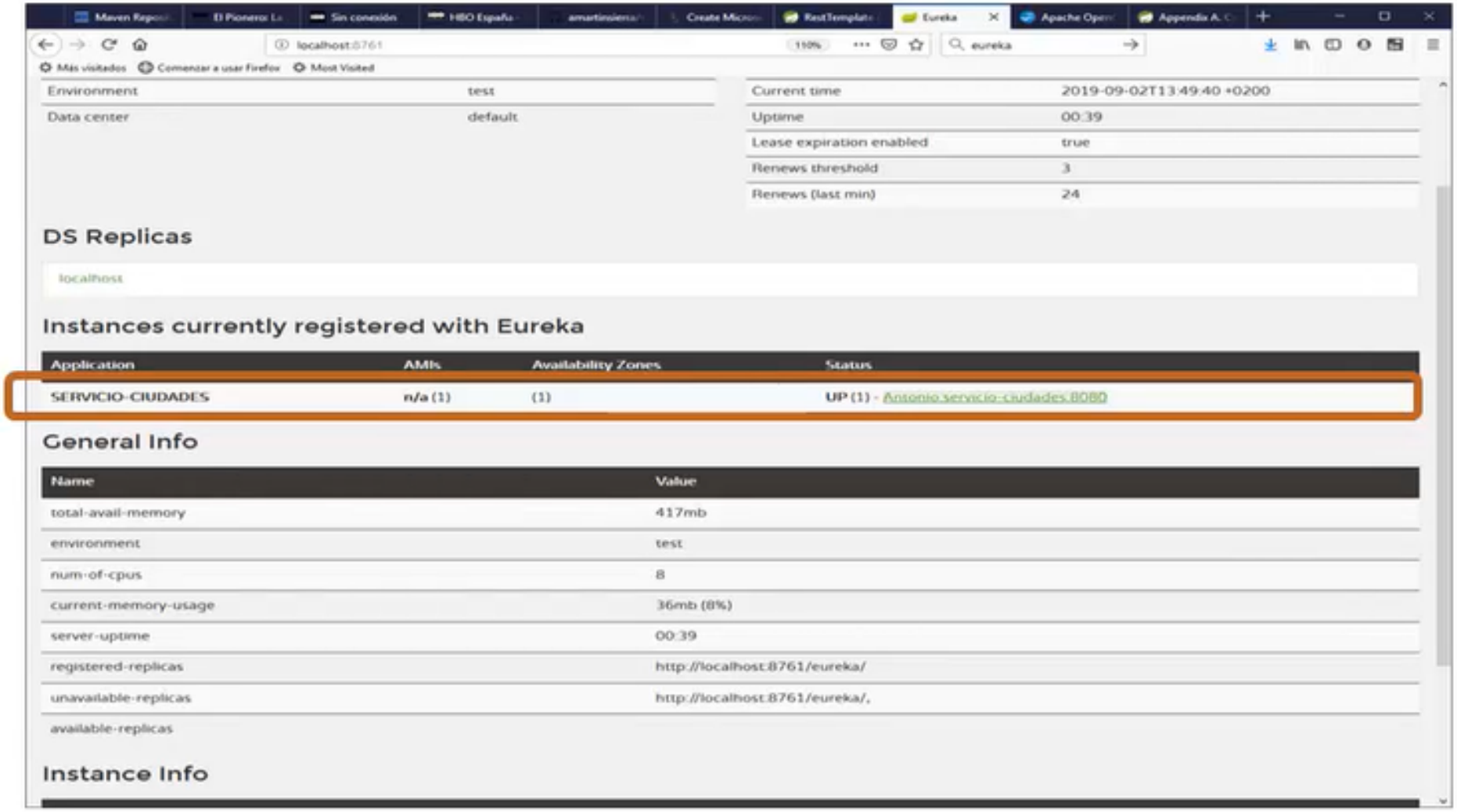
- Si se utiliza **application.yml** habrá que añadir:

```
spring:
  application:
    name: identificadorespecificodentrodeeureka
eureka:
  client:
    service-url:
      defaultZone: http://localhost:8761/eureka
```

Ejemplo microservicio de libros

- Ahora necesitamos el starter **Eureka Discovery Client**
- (proyecto>botón derecho>Spring >Add Starters)
- **Configuraciones**
- spring.application.name=**servicio-libros**
- eureka.client.service-url.defaultZone=http://localhost:8761/eureka

http://localhost:8761



Environment	test	Current time	2019-09-02T13:49:40 +0200
Data center	default	Uptime	00:39
		Lease expiration enabled	true
		Renews threshold	3
		Renews (last min)	24

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
SERVICIO-CIUDADES	n/a (1)	(1)	UP (1) - Antonio:servicio-ciudades:8080

General Info

Name	Value
total-avail-memory	417mb
environment	test
num-of-cpus	8
current-memory-usage	36mb (8%)
server-uptime	00:39
registered-replicas	http://localhost:8761/eureka/
unavailable-replicas	http://localhost:8761/eureka/,
available-replicas	

Instance Info

Descubrimiento de servicios en Eureka

- Requiere dependencia **Eureka Discovery Client**

```
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
```

- Anotar el método de creación RestTemplate con @LoadBalanced para activar la librería **Ribbon**

```
@LoadBalanced
@Bean
public RestTemplate template() {
return new RestTemplate();
}
```

Descubrimiento de servicios en Eureka

- La librería **Ribbon** es la que se encarga de conectar con Eureka utilizar el identificador del microservicio y descubrir en qué dirección real está ese microservicio.

- Acceso mediante el nombre de registro:
  - String URL_SERVICE = “http://i**dentificadorespecificodentrodeeureka**”;

Descubrimiento de servicios en Eureka

- **Configuraciones**
- En **application.properties** habrá que añadir:

```
spring.application.name=identificadorNombreServicio
eureka.client.service-url.defaultZone=http://localhost:8761/eureka
eureka.client.register-with-eureka=false
```

- Si se utiliza **application.yml** habrá que añadir:

```
eureka:
  client:
    register-with-eureka: false       #El cliente no necesita ser descubierto
    service-url:
      defaultZone: http://localhost:8761/eureka
spring:
  application:
    name: cliente-libros
```

Ejemplo microservicio cliente crud libros bd

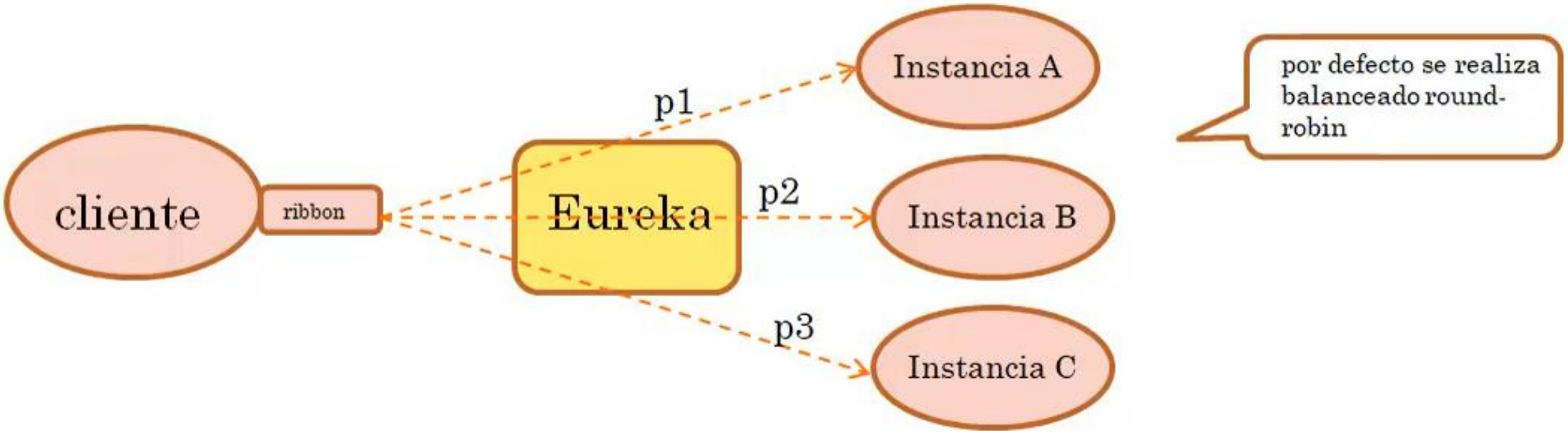
- Añadir el starter (**Eureka Diskovery Client**)
- En configuraciones:
- Activar librería Ribbon añadiendo @LoadBalanced (en Application.java)
- En application.properties añadir
- spring.application.name=cliente-libros
- eureka.client.service-url.defaultZone=http://localhost:8761/eureka
- eureka.client.register-with-eureka=false
- Modificar el código (cambiar http://localhost:8080 por http://servicio-libros)

Múltiples instancias de un servicio

- Tener múltiples instancias de un servicio proporciona beneficios como una alta disponibilidad o una mejora del rendimiento.

- **Balanceado de peticiones**
- La librería Ribbon es capaz de balancear peticiones a un microservicio entre varias instancias del mismo.

Múltiples instancias de un servicio



- Este balanceado permite alta disponibilidad, si una instancia falla están otras dando servicio y mejora del rendimiento, si están desplegadas en diferentes máquinas las peticiones se van a repartir entre ellas.

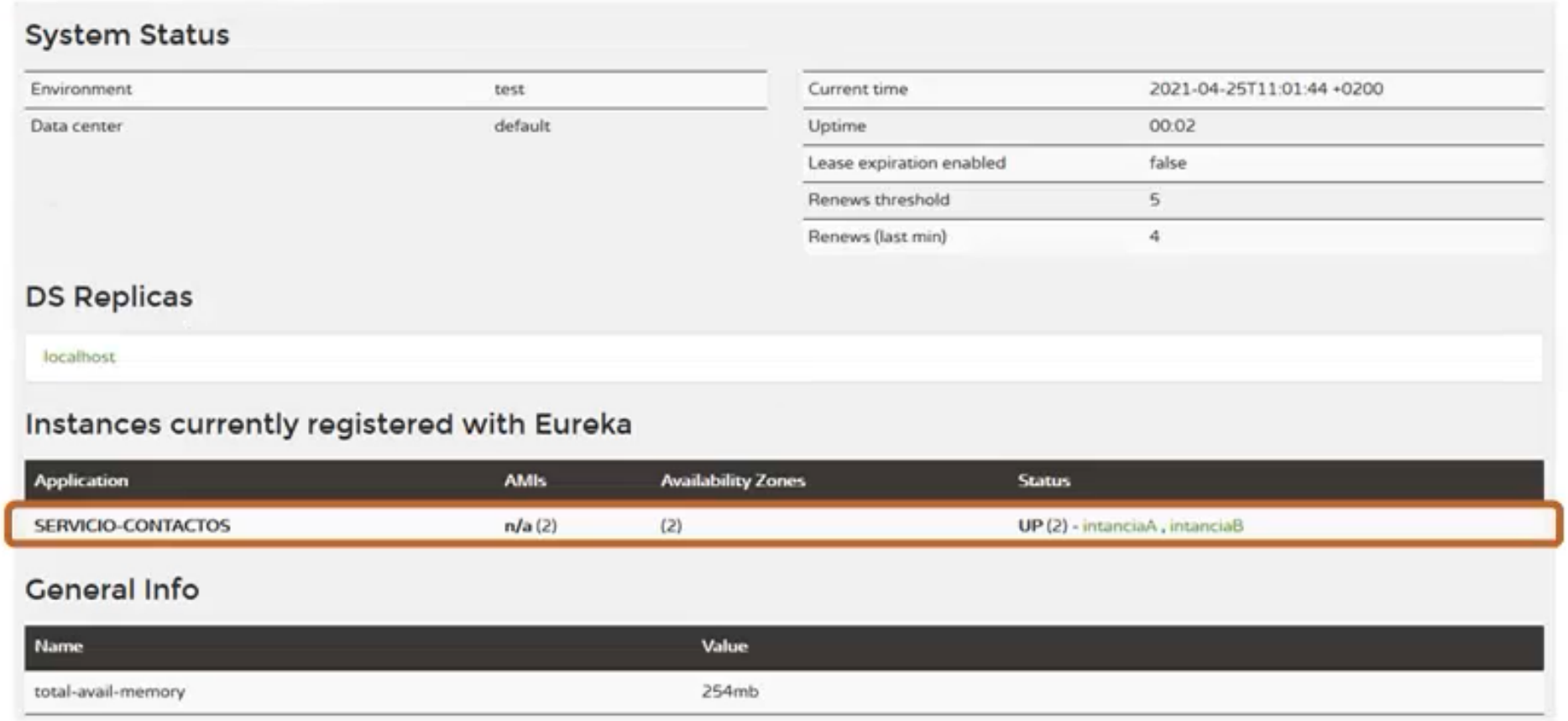
Configuración del servicio

- Todas las instancias se pueden generar desde la misma aplicación Spring Boot

- En el **application.properties** cada instancia del servicio deberá tener un valor único para la propiedad **eureka.instance.instanceId** con el nombre de cada instancia
- **Con valor fijo:**
- eureka.instance.instance-id=instancia1
- **Con valor aleatorio:**
- eureka.instance.instance-id=${random.value}:${spring.application.name}

Registro en Eureka

- Al lanzar cada instancia del mismo servicio, aparecerán en Eureka:



System Status

Environment	test	Current time	2021-04-25T11:01:44 +0200
Data center	default	Uptime	00:02
		Lease expiration enabled	false
		Renews threshold	5
		Renews (last min)	4

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
SERVICIO-CONTACTOS	n/a (2)	(2)	UP (2) - intanciaA , intanciaB

General Info

Name	Value
total-avail-memory	254mb