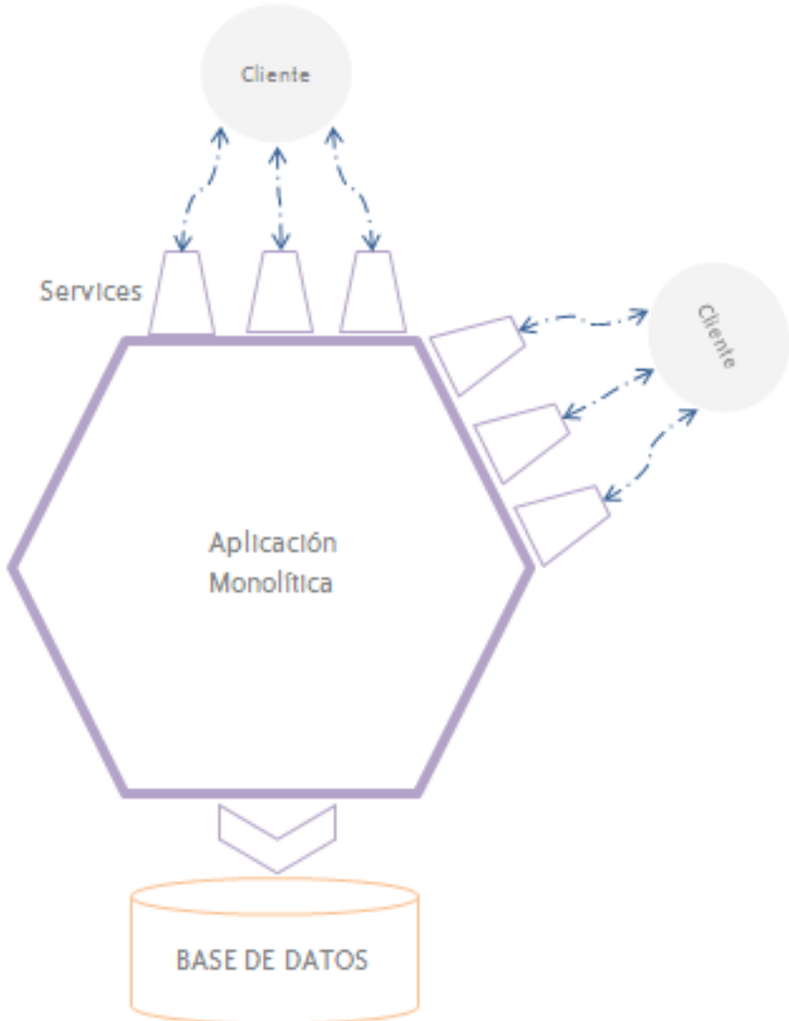
MAINT

Microservicios

Día 4

Microservices, Modelos de Implementación

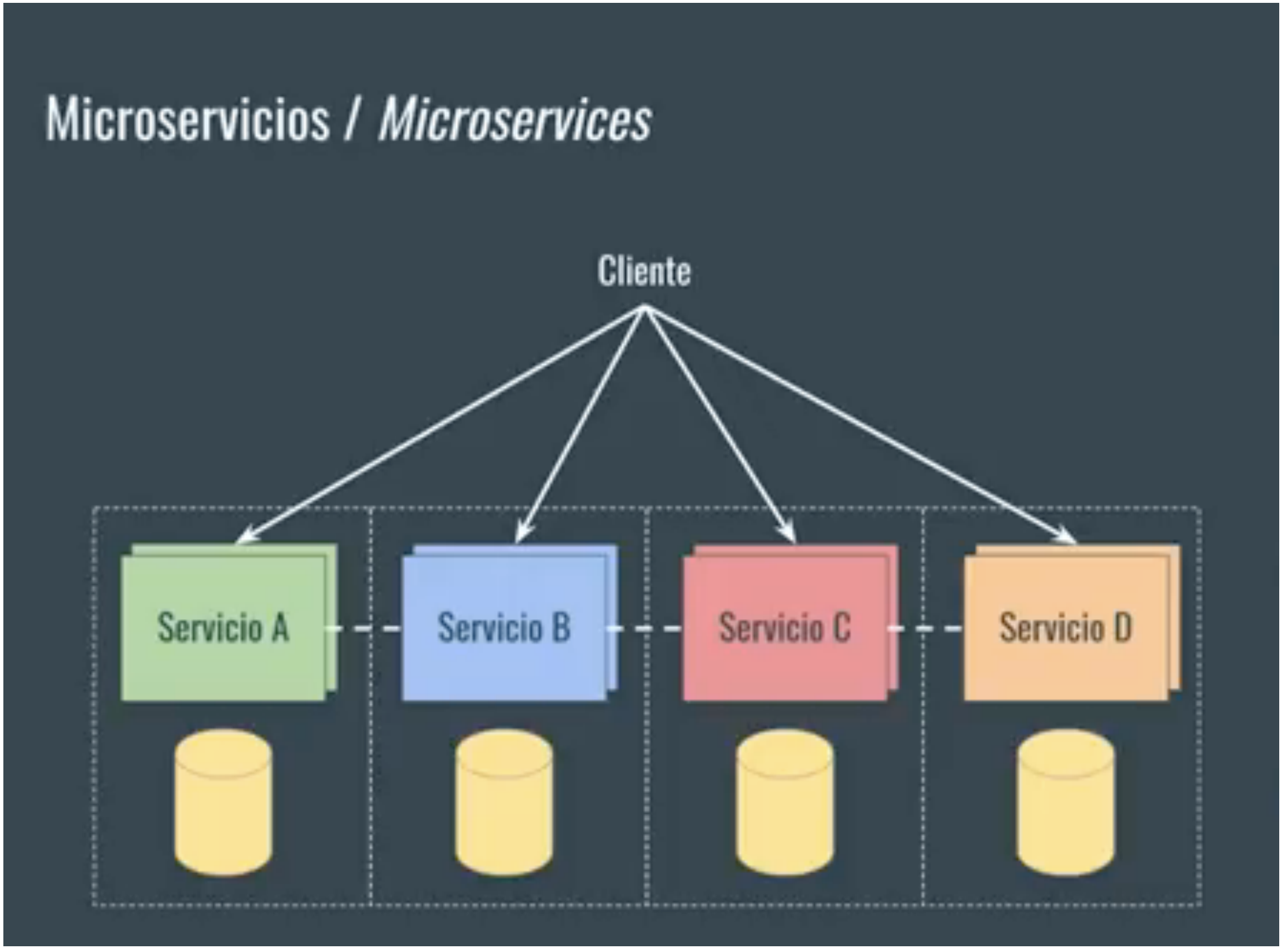
Historia


Cliente
Services
Cliente
Aplicación
Monolítica
BASE DE DATOS

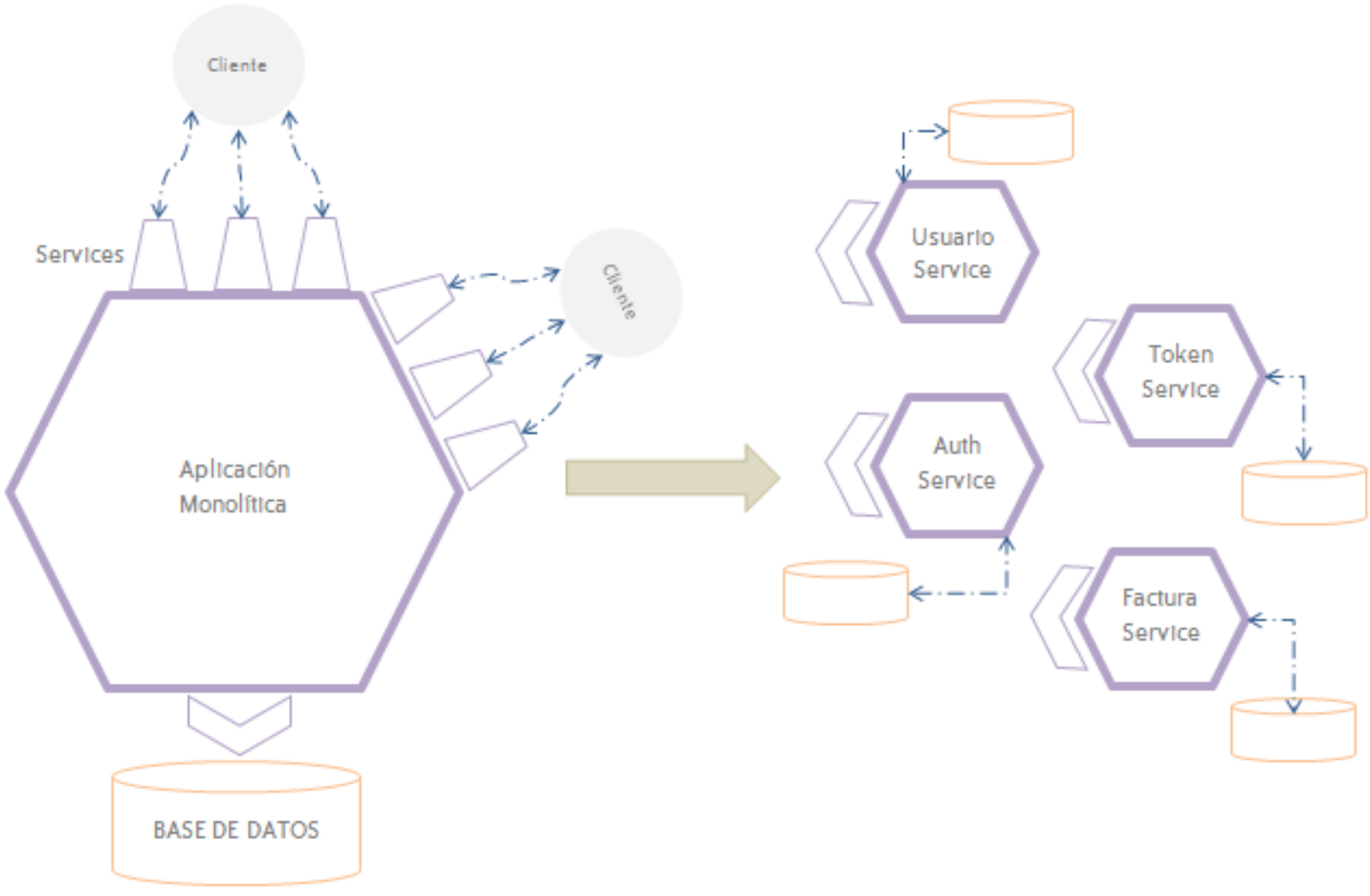
Microservicios, clave para el desarrollo de aplicaciones y servicios.

La base de la arquitectura de microservicios o MSA, se basa en el desarrollo de una única aplicación como un conjunto de grano fino y servicios independientes, encargándose cada microservicio de su propio proceso, desarrollo, y despliegue de forma independiente.

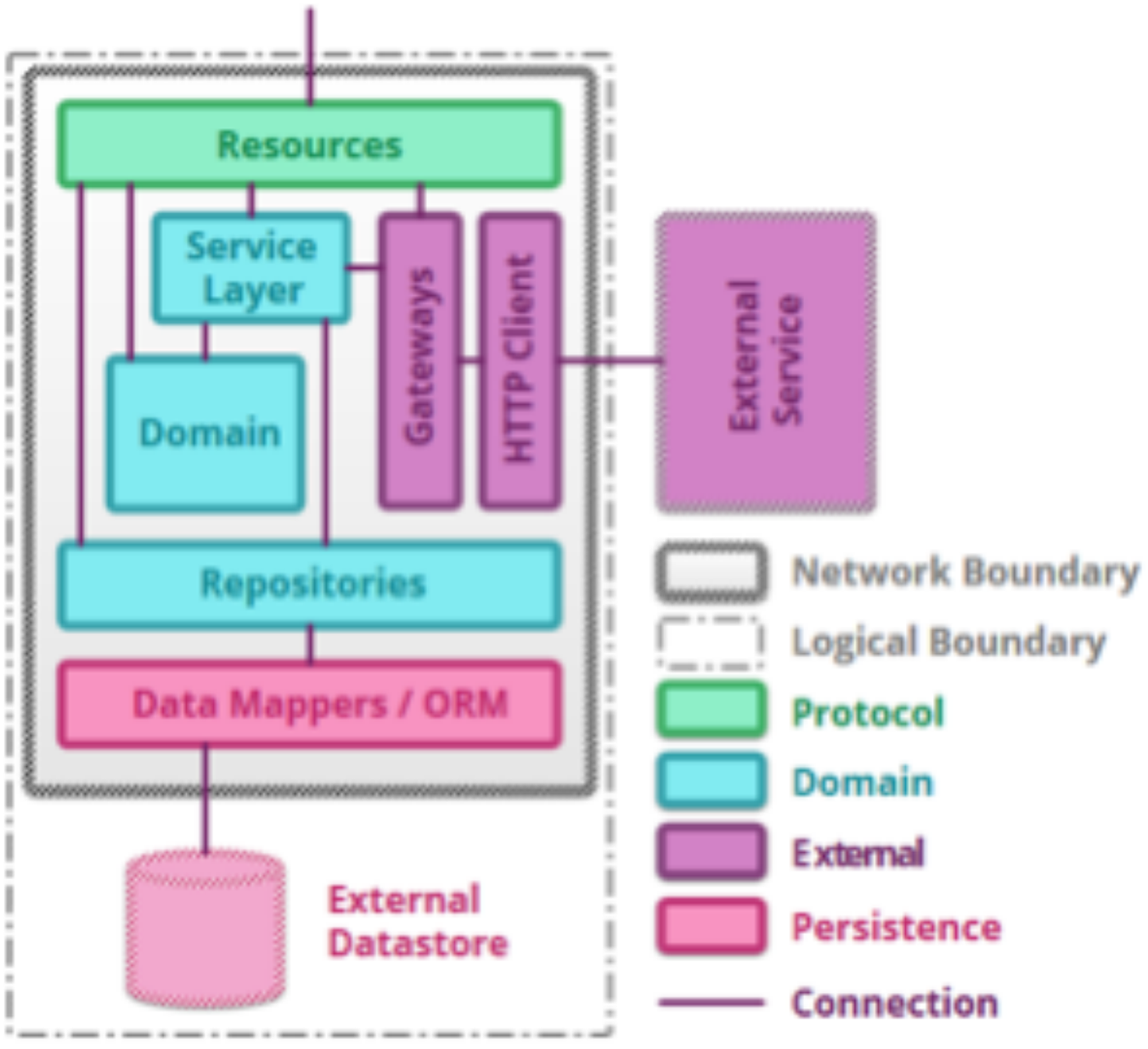
Los microservicios han ido evolucionando con el tiempo, y debido a ello, en la actualidad la lógica del negocio y la comunicación de red está dispersa en servicios independientes, lo que supone la eliminación del ESB central.


Microservicios / *Microservices*
Cliente
Servicio A
Servicio B
Servicio C
Servicio D

Qué es un microservicio

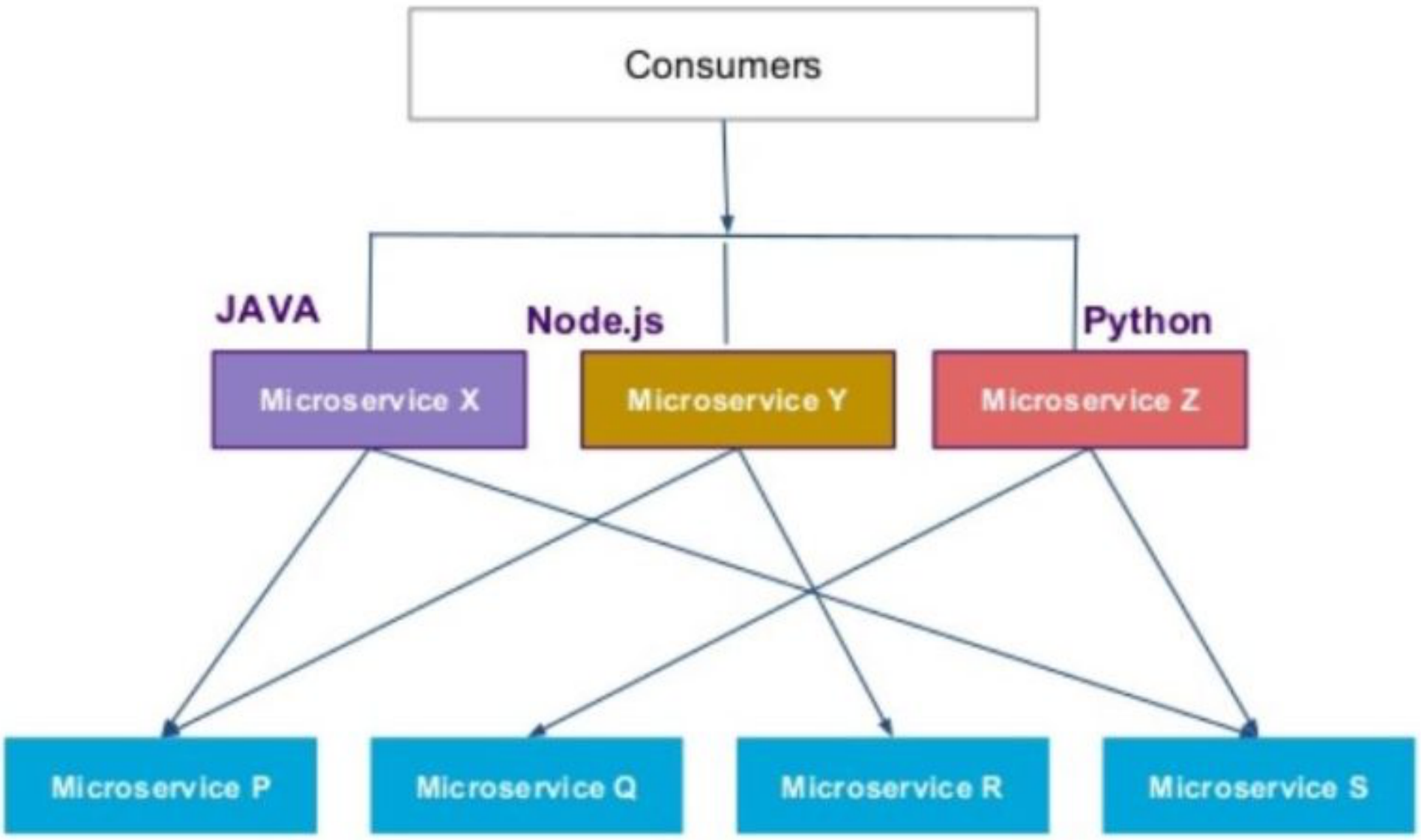


Qué es un microservicio



Microservicios, clave para el desarrollo de aplicaciones y servicios.

Los microservicios dependen en gran medida de la comunicación entre los mismos, a lo que añadía que existen varios patrones de comunicación, como el envío de mensajes sincronizados basados en protocolos muy conocidos y ampliamente adoptados (HTTP). Además, es importante destacar que la mensajería asincrónica basada en protocolos abiertos como MQTT o AMQP, es clave para fomentar la autonomía de los microservicios.


Consumers
JAVA
Node.js
Python
Microservice X
Microservice Y
Microservice Z
Microservice P
Microservice Q
Microservice R
Microservice S

“Con MSA, la lógica del negocio y la lógica de la comunicación de red, están dispersas a través de servicios independientes”.

Microservicios, clave para el desarrollo de aplicaciones y servicios.

Cuantos más microservicios se desplieguen y operen, más se deberá tener en cuenta el espacio de recursos individuales de cada microservicio.

Aquellos aspectos que deben tenerse en consideración en el momento de implantar una MSA son:

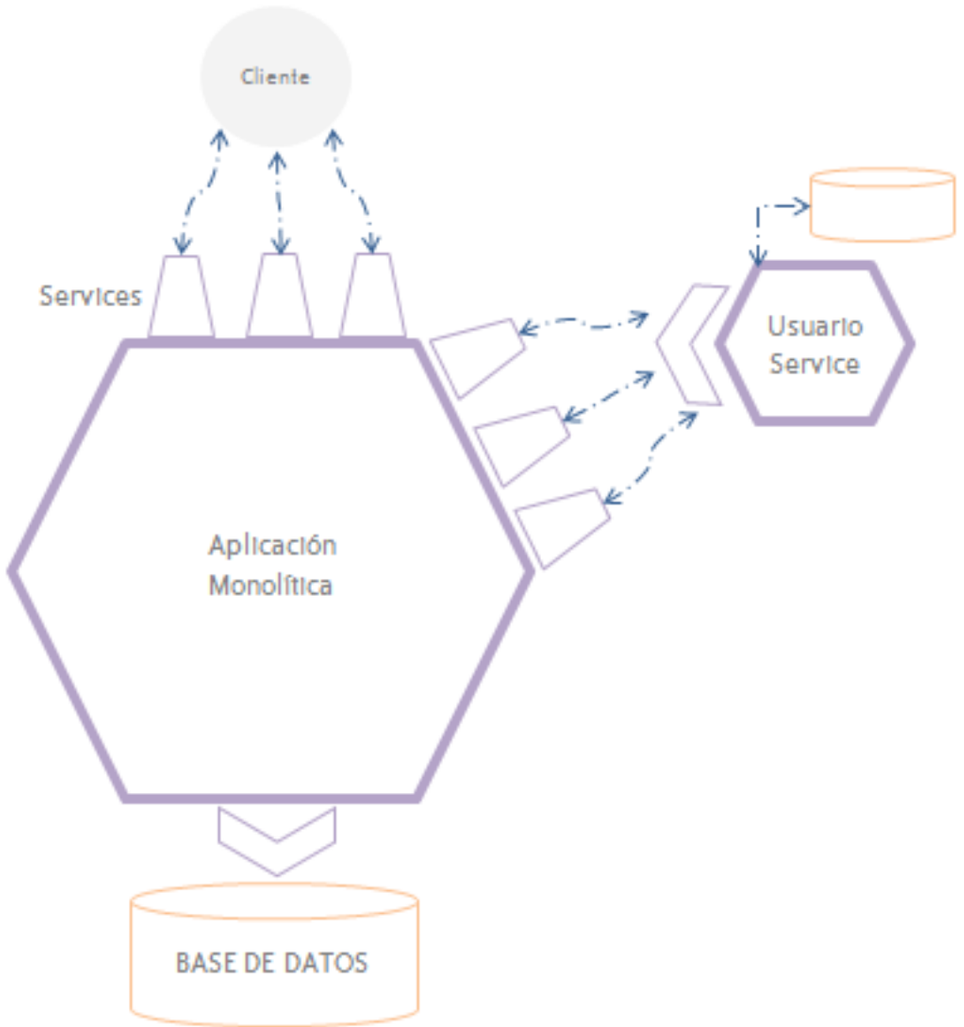
- Estrategias de implementación de microservicios.
- El papel del diseño dirigido por dominio en la arquitectura de microservicios.
- Asegurar microservicios (Oauth2, OpenIDI Connect y JWT).
- Granularidad de microservicios y su organización.
- Varios registros de servicios, ETCD.

Microservicios, clave para el desarrollo de aplicaciones y servicios.

También existe la posibilidad de crear microservicios compuestos mediante la integración de API web / SaaS, sistemas heredados y microservicios. El servicio API o Edge Service también es un microservicio de integración con algunas capacidades de puerta de enlace API.

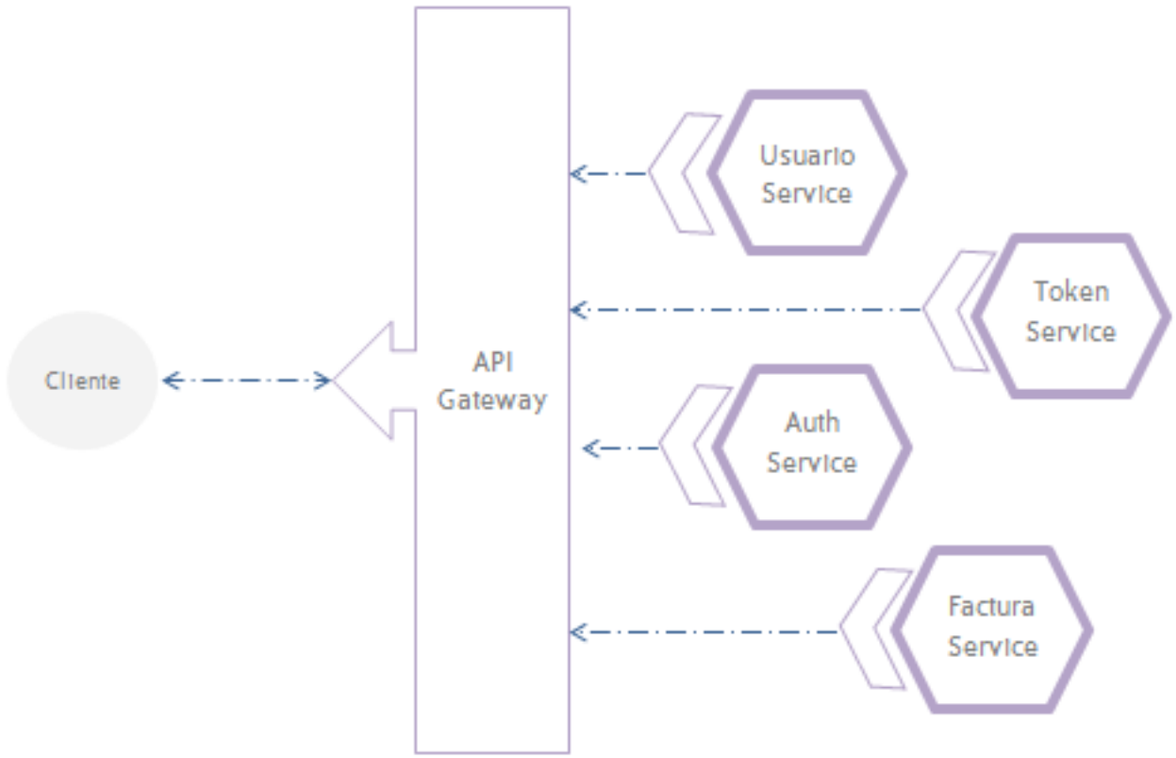
Tres aspectos importantes: la posibilidad de eliminar ESB centralizado con la arquitectura de microservicios, la importancia de crear servicios compuestos utilizando tecnología dedicada y la integración de microservicios con Ballerina.

Aplicaciones monolíticas

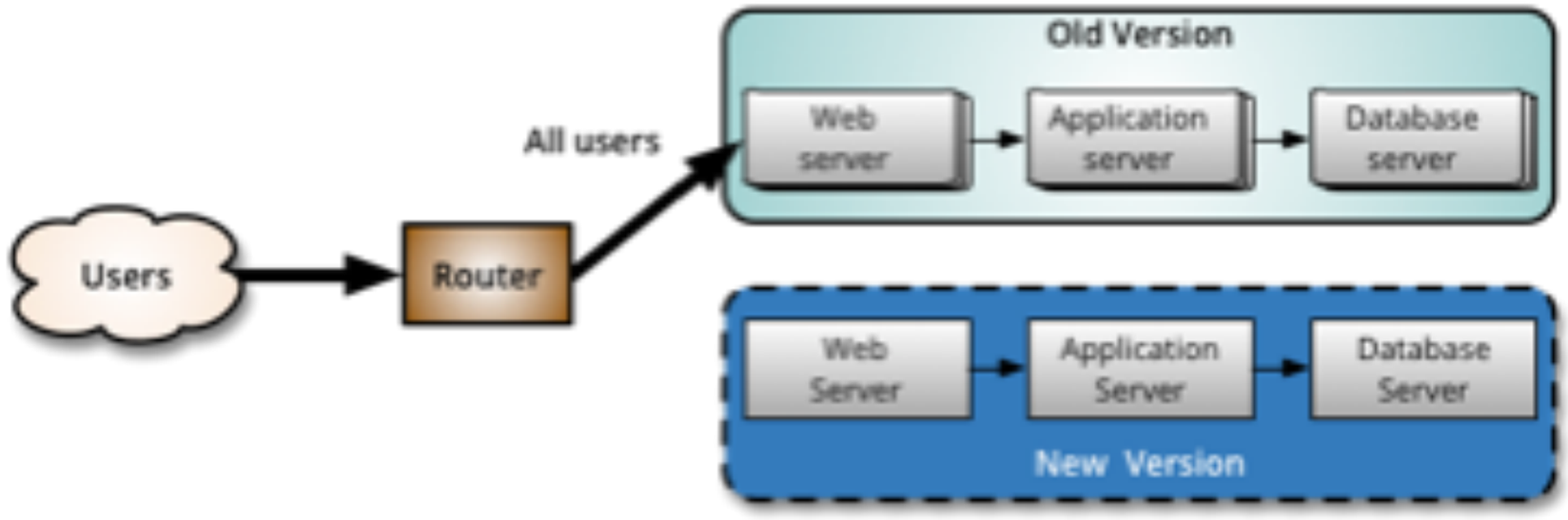


MAINT

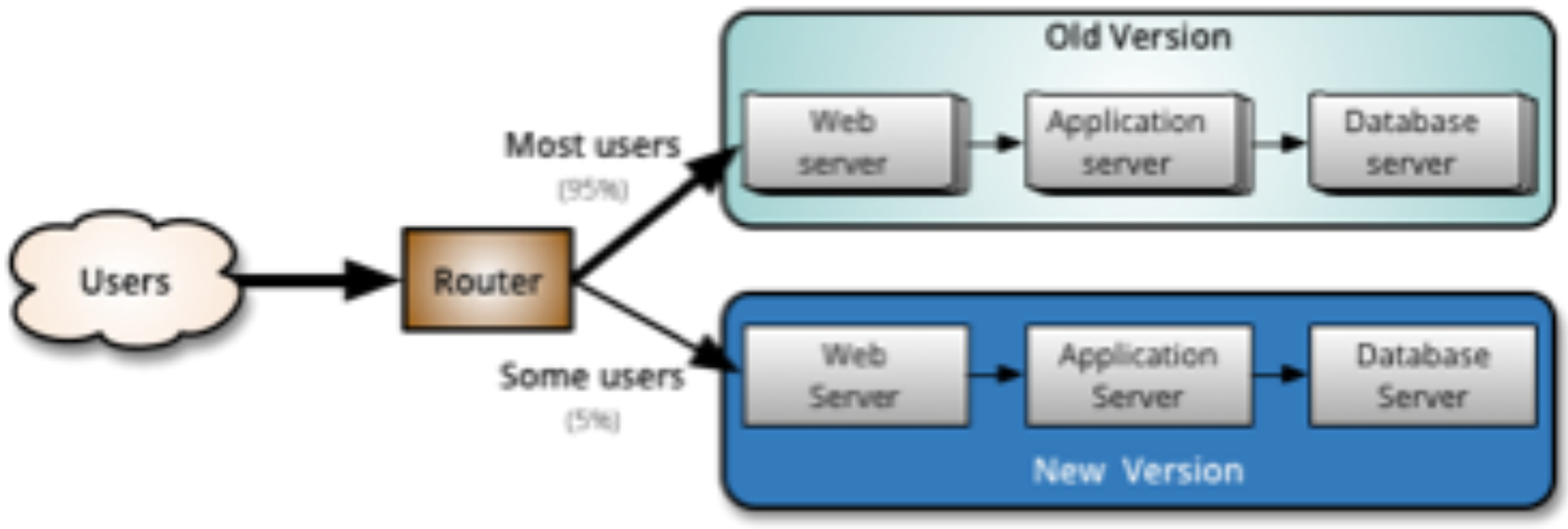
Microservices y un API Gateway



Arquitecturas Canary: Microservicios y ESB trabajando juntos

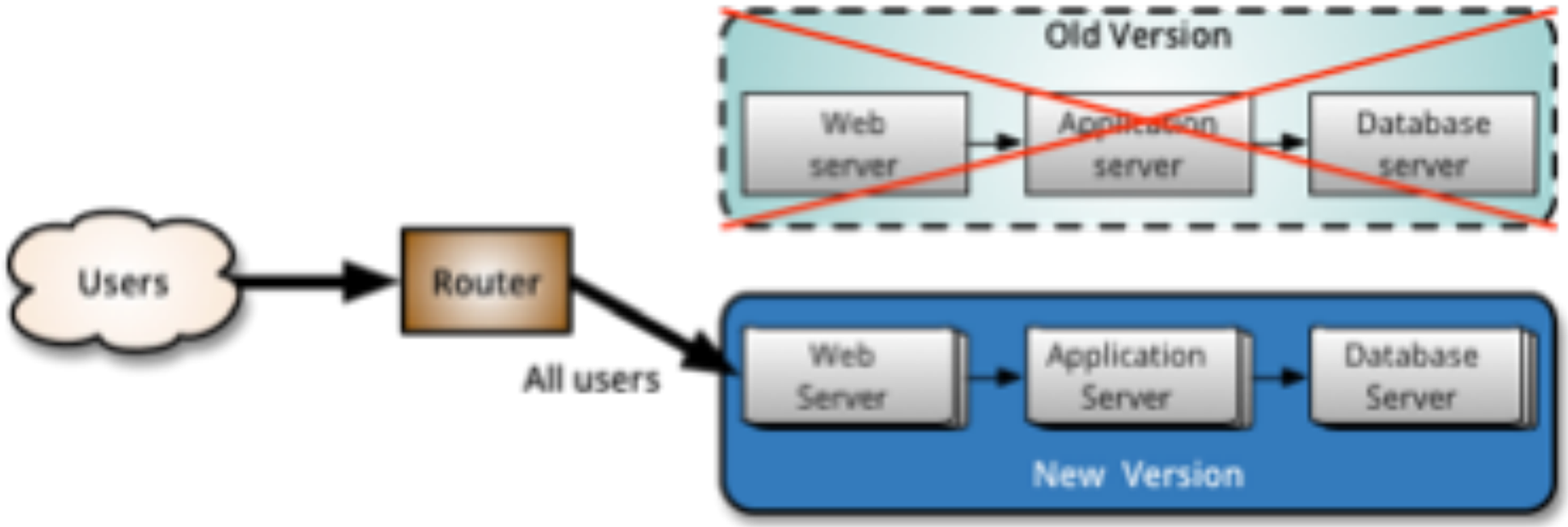


Arquitecturas Canary: Microservicios y ESB trabajando juntos



MAINT

Arquitecturas Canary: Microservicios y ESB trabajando juntos



MAINT

Los microservicios suponen un nuevo estilo de arquitectura software para el diseño de aplicaciones.

Pero, ¿qué son los microservicios?

- Según Martin Fowler y James Lewis explican en su artículo Microservices, los microservicios se definen como un estilo arquitectural, es decir, una forma de desarrollar una aplicación, basada en un conjunto de pequeños servicios, cada uno de ellos ejecutándose de forma autónoma y comunicándose entre si mediante mecanismos livianos, generalmente a través de peticiones REST sobre HTTP por medio de sus APIs.

Pero, ¿qué son los microservicios?

- La tendencia es que las aplicaciones sean diseñadas con un enfoque orientado a microservicios, construyendo múltiples servicios que colaboran entre si, en lugar del enfoque monolítico, donde se construye y despliega una única aplicación que contenga todas las funcionalidades.

Algunas de las características más destacadas de los microservicios son:

- Pueden ser auto-contenidos, de tal forma que incluyen todo lo necesario para prestar su servicio

- Servicios pequeños, lo que facilita el mantenimiento. Ej: Personas, Productos, Posición Global, etc

Algunas de las características más destacadas de los microservicios son:

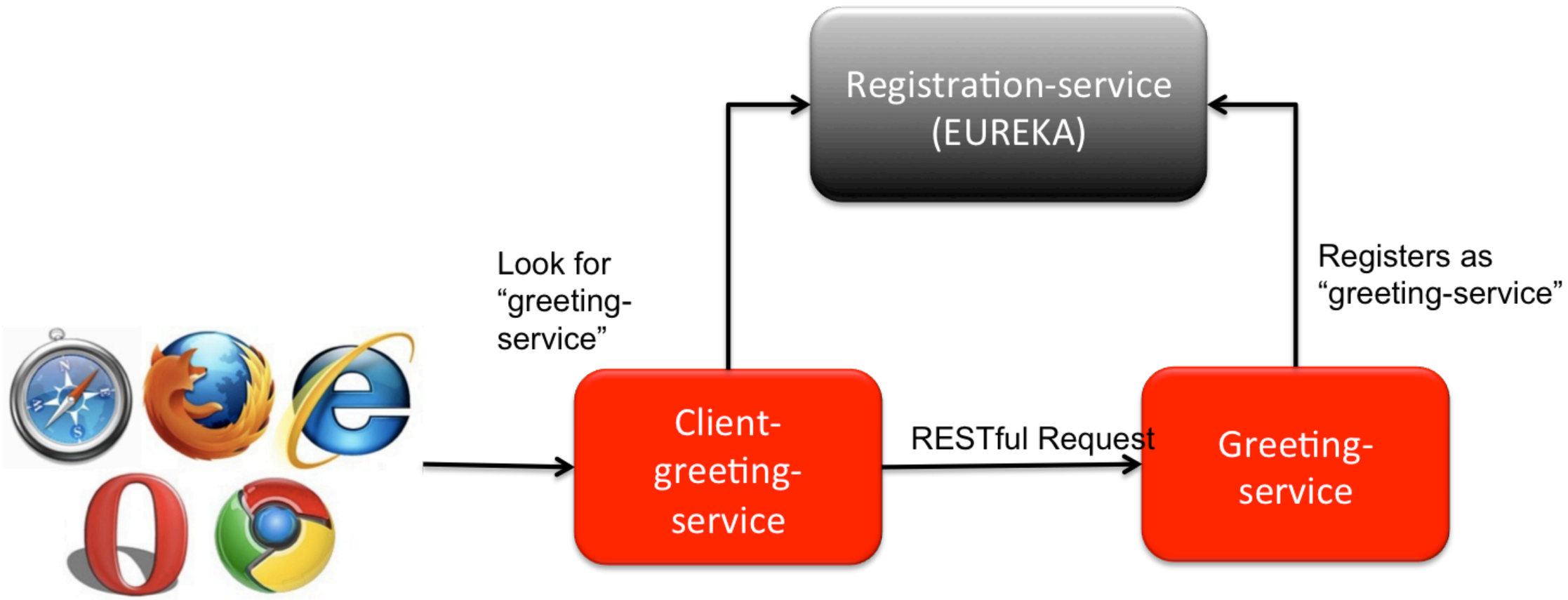
- Principio de responsabilidad única: cada microservicio hará una única cosa, pero la hará bien

- Políglotas: una arquitectura basada en microservicios facilita la integración entre diferentes tecnologías (lenguajes de programación, BBDD…etc)

Algunas de las características más destacadas de los microservicios son:

- Despliegues unitarios: los microservicios pueden ser desplegados por separado, lo que garantiza que cada despliegue de un microservicio no implica un despliegue de toda la plataforma. Tienen la posibilidad de incorporar un servidor web embebido como Tomcat o Jetty

- Escalado eficiente: una arquitectura basada en microservicios permite un escalado elástico horizontal, pudiendo crear tantas instancias de un microservicio como sea necesario.



Eureka: Hello World – Cómo construir microservicios con Spring Boot

Laboratorio 7

HYSTRIX

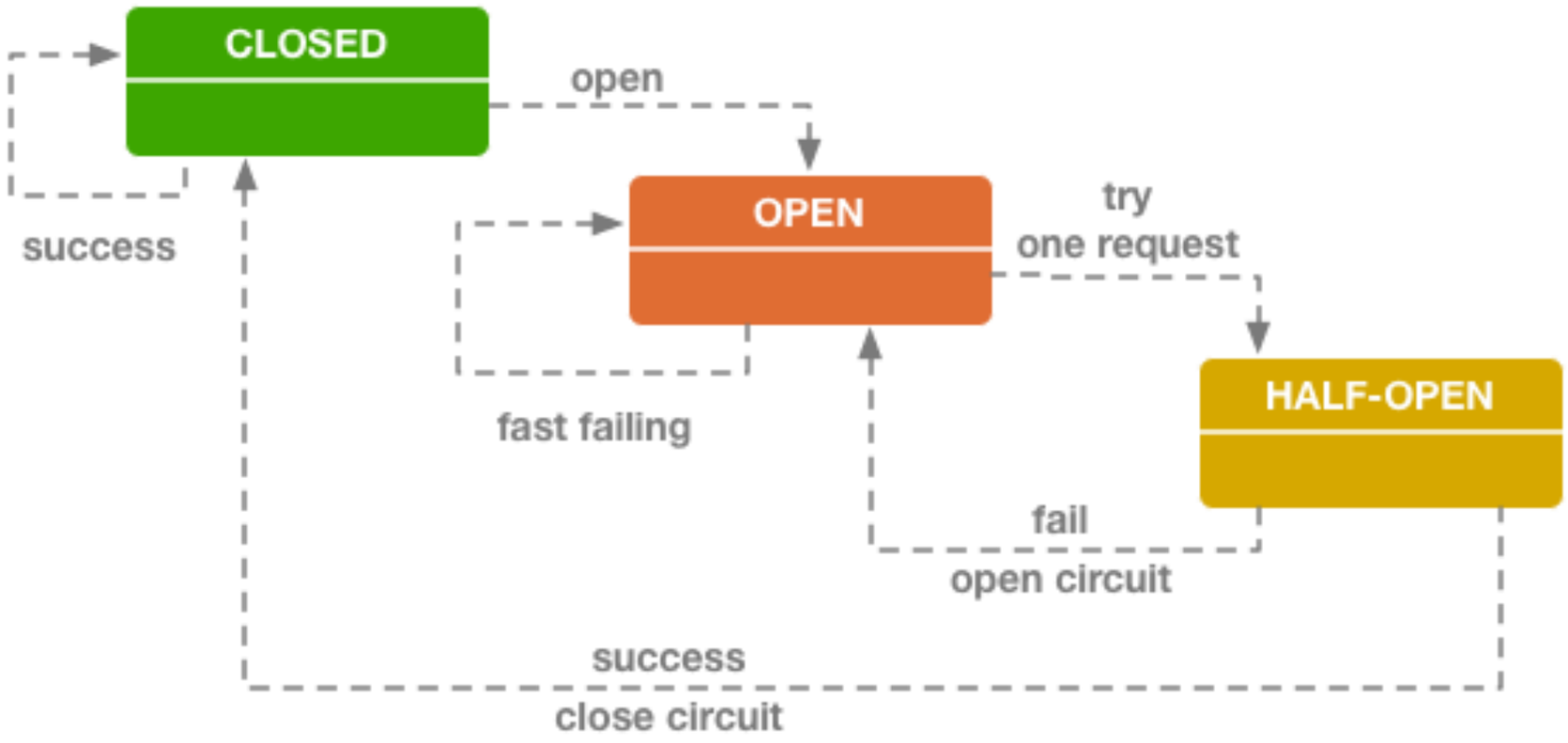
- La comunicación entre sistemas mediante llamadas remotas es un procedimiento muy habitual hoy en día. Este tipo de comunicaciones, si cabe, cobra aun más relevancia en sistemas basados en arquitecturas orientadas a microservicios.

- El problema de las comunicaciones remotas es que pueden fallar (servicios caídos, demasiados consumidores realizando peticiones generando mayor lentitud en las respuestas, etc). Patrones de diseño como Circuit Breaker permiten establecer un mecanismo de control y gestión de los problemas derivados de la comunicación remota, ayudando, por tanto, a mejorar la resiliencia de los sistemas

¿Qué es Hystrix?

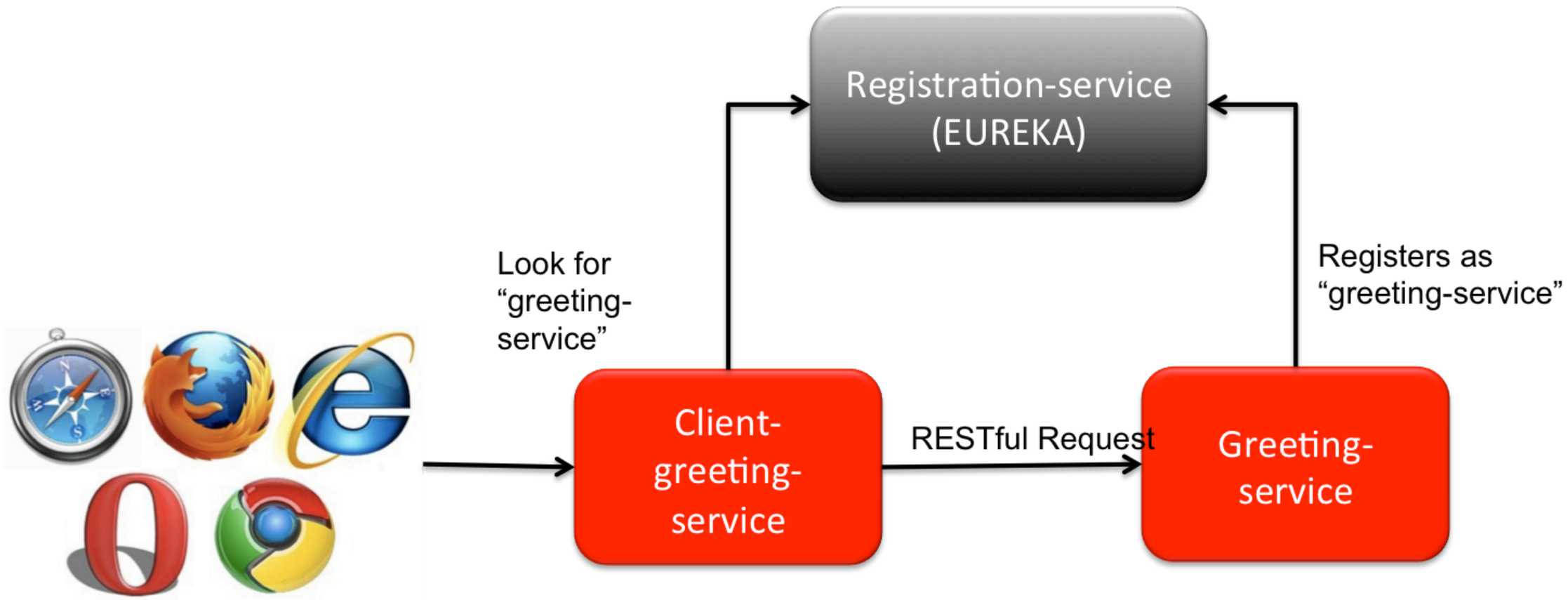
- Se trata de una librería ofrecida por Netflix diseñada para aislar puntos de acceso a sistemas remotos, servicios y librerías de terceros, deteniendo fallos en cascada y permitiendo mejorar la resiliencia en sistemas complejos distribuidos donde la probabilidad de fallo es inevitable.

El siguiente diagrama de estados muestra como un sistema resiliente funciona durante el ciclo de vida del circuit breaker



Circuit Breaker State Diagram

- Si el sistema funciona con normalidad, el circuit breaker permanecerá cerrado (estado closed). En el caso de comenzar a ser inestable y se alcance un limite de fallos, el circuito se abrirá para prevenir más errores (estado open).
- Cada cierto intervalo de tiempo, se pasa al estado half open donde Hystrix se encarga de enviar una primera petición para comprobar la disponibilidad del sistema, pasando de nuevo al estado closed (si el funcionamiento vuelve a ser el correcto) u open (si la inestabilidad del sistema continua).



Hystrix: Implementa un circuit breaker – Cómo construir microservicios con Spring Boot

Laboratorio 8

MAINT

Gracias