Comunicación entre Microservicios

MitoCode Network

Por: Andres Gonzales

MitoCode

Objetivos

- Entender los tipos de comunicación en microservicios.
- Diferenciar **sincronía vs asincronía**.
- Implementar comunicación REST con **WebClient** y **Feign**.
- Conocer gRPC como alternativa.
- Preparar la base para patrones avanzados como **SAGA** y mensajería con Kafka

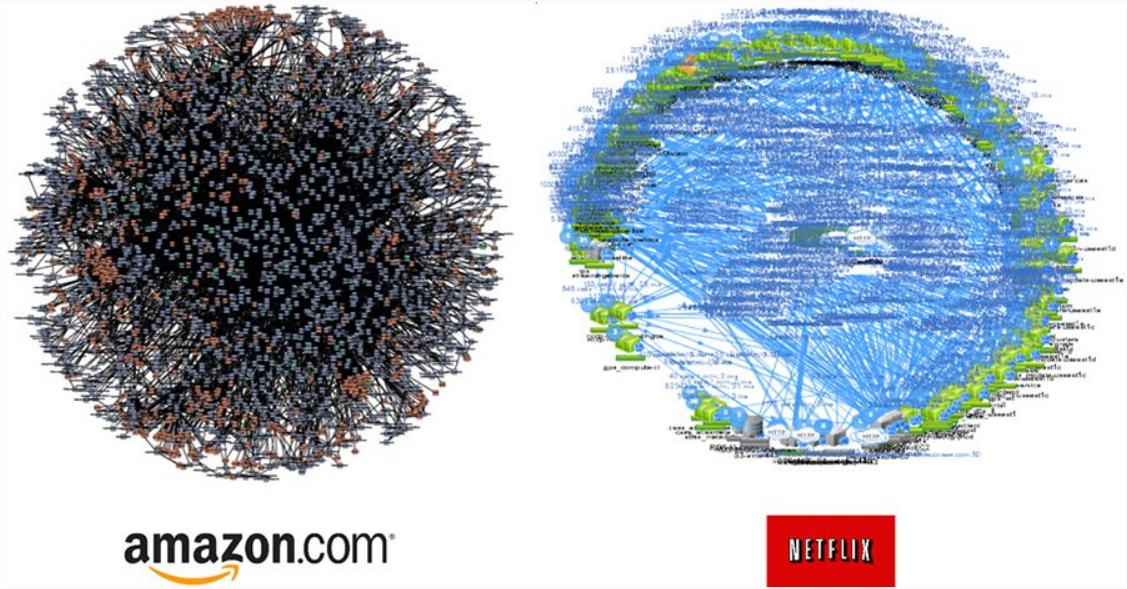
MitoCode

Por qué la Comunicación es Crítica

En una arquitectura distribuida:

- Los microservicios rara vez operan aislados.
- Una petición de usuario puede requerir datos de varios servicios.
- Debemos cuidar **rendimiento, resiliencia y acoplamiento**.

Por qué la Comunicación es Crítica



MitoCode

Tipos de Comunicación: Síncrona

Ejemplo: REST API, gRPC

Ventajas: Simplicidad, respuesta inmediata

Desventajas: Acoplamiento temporal, afecta disponibilidad

Tipos de Comunicación: Asíncrona

Ejemplo: Kafka, RabbitMQ

Ventajas: Desacople temporal, resiliencia

Desventajas: Complejidad, posible latencia

Comunicación Síncrona

- Ideal para operaciones que requieren respuesta inmediata.
- Puede ser **REST** (más común) o **gRPC** (más eficiente).
- Riesgo: si un servicio falla o se ralentiza, afecta a toda la cadena.

Comunicación Asíncrona

- Ideal para procesos desacoplados o eventos de dominio.
- Uso de **eventos** y **colas de mensajes**.
- Alta resiliencia y escalabilidad, pero no apta para todo caso.

Consideraciones y Buenas Prácticas

- Manejar timeouts y reintentos.
- Usar nombres lógicos con Eureka en lugar de IPs fijas.
- Evitar llamadas en cascada muy profundas.
- Considerar asíncrono cuando no se necesita respuesta inmediata.

MitoCode Network

www.mitocode.com