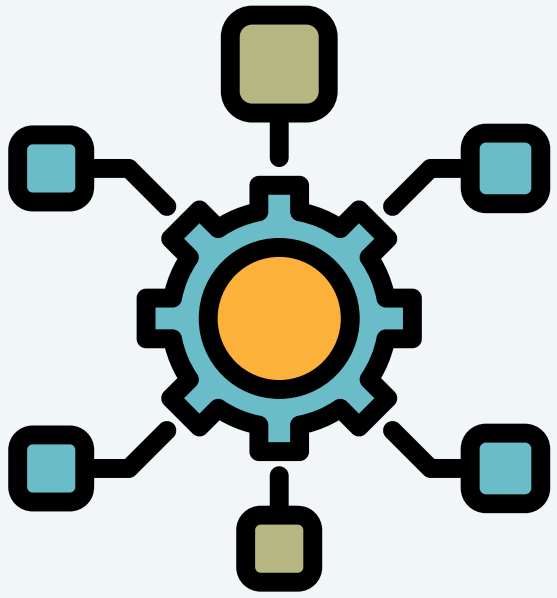
Gestión de Errores en Microservicios

Guía con todas las claves y principales estrategias, para crear arquitecturas robustas, a prueba de errores



PABLO DEL ÁLAMO

Introducción

Los errores en microservicios pueden surgir por múltiples razones: fallos en la red, errores de servidor, fallos en el código....

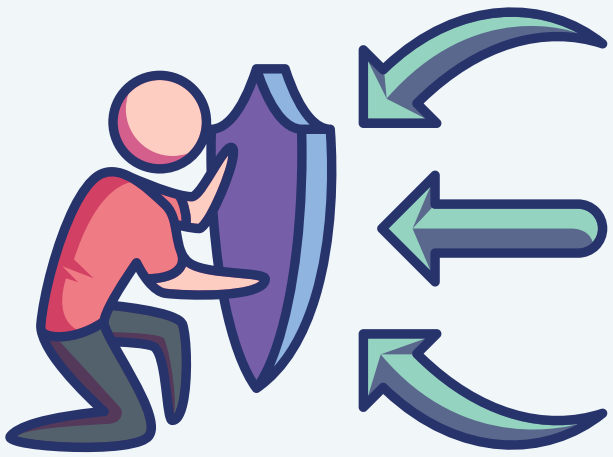
En esta guía vamos a ver cómo gestionar estos fallos, para no cargarnos el sistema o aplicación por completo, y asegurar que sigan operativos incluso después de estos errores.

PABLO DEL ÁLAMO

Principio Fundamental: Resiliencia del Sistema

En un entorno de microservicios, la resiliencia significa que, incluso si un microservicio falla, el sistema sigue funcionando de manera aceptable.

Buscamos crear una arquitectura preparada contra todo tipo de fallos.



PABLO DEL ÁLAMO

Técnicas de Gestión de Errores

Hablemos de las principales técnicas: Retries, Circuit Breaker, Timeout y Fallbacks.

Todas ellas son muy útiles para mitigar el impacto de un error en un microservicio.

PABLO DEL ÁLAMO

Técnica 1: Retries

Retry es volver a intentar una operación fallida.

Ideal cuando un error podría resolverse de un momento a otro por sí solo.

¡Pero ojo! No abuses de esto, puedes ocasionar una carga innecesaria en el sistema.

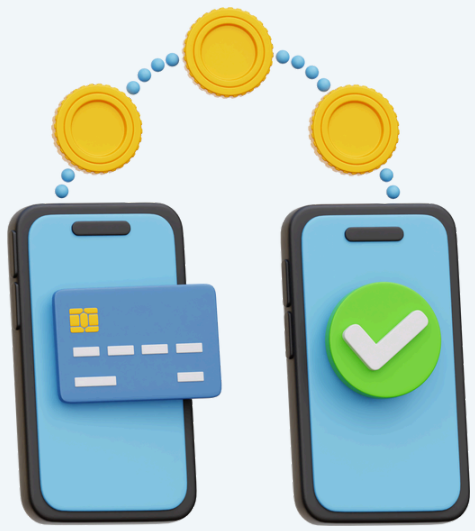
PABLO DEL ÁLAMO

Ejemplo de Uso de Retries

Imagina que un servicio de pago no responde.

Un retry puede salvar la transacción al conseguir que se complete al segundo intento.

Perfecto para errores transitorios.

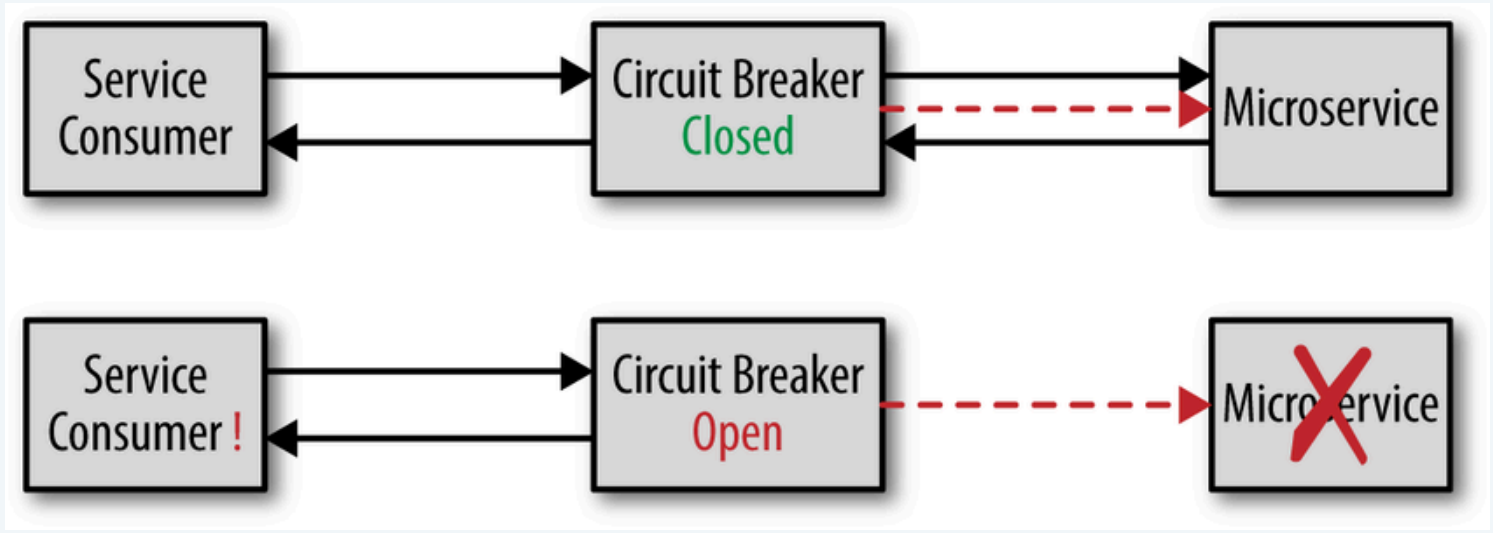


PABLO DEL ÁLAMO

Técnica 2: Circuit Breaker

Funciona como un interruptor, abriéndose (bloqueando llamadas) si un servicio sigue fallando tras varios intentos.

Así se evita sobrecargar un servicio ya fallido.

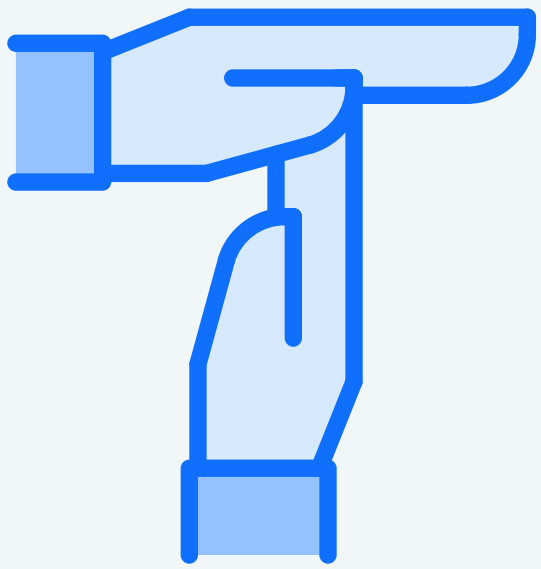


PABLO DEL ÁLAMO

Técnica 3: Timeout

Configura un tiempo máximo para esperar por una respuesta.

Previene que tu aplicación quede atrapada en un bucle esperando indefinidamente.



PABLO DEL ÁLAMO

Técnica 4: Fallbacks de API

Usa un método alternativo cuando el principal falla.

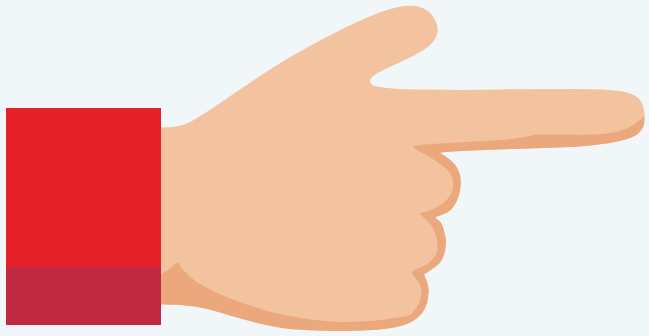
Por ejemplo, si un servicio de recomendaciones no está disponible, ofrece productos populares en su lugar.

PABLO DEL ÁLAMO

Gestión de Dependencias Cruzadas

Las dependencias cruzadas pueden convertir errores en un dominó de fallos.

Usa estas técnicas para contener el problema.



PABLO DEL ÁLAMO

Ejemplo de Gestión de Dependencias

Si el servicio de autenticación falla, otros servicios pueden devolver un modo “degradado” que permita operaciones básicas.

PABLO DEL ÁLAMO

Herramientas de Monitorización

Usa herramientas como Prometheus o Grafana para visualizar en tiempo real lo que sucede en tus servicios.

PABLO DEL ÁLAMO

Ejemplo de Monitoreo Visual

PABLO DEL ÁLAMO

Circuit Breaker Avanzado: Hystrix

Netflix Hystrix es una biblioteca popular para implementar circuit breakers en aplicaciones Java.

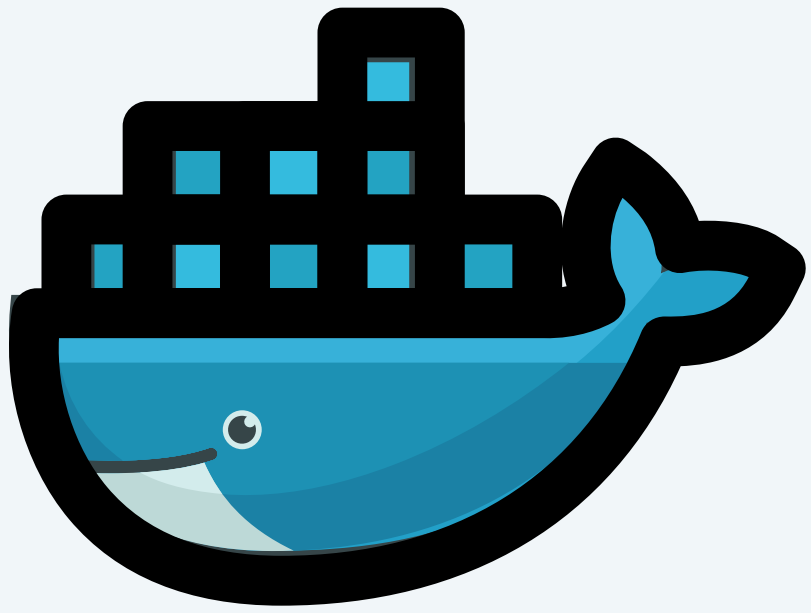
Investiga cómo puede integrarse en tu arquitectura.

PABLO DEL ÁLAMO

Docker y Kubernetes

La contenerización y orquestación pueden limitar el impacto de los errores.

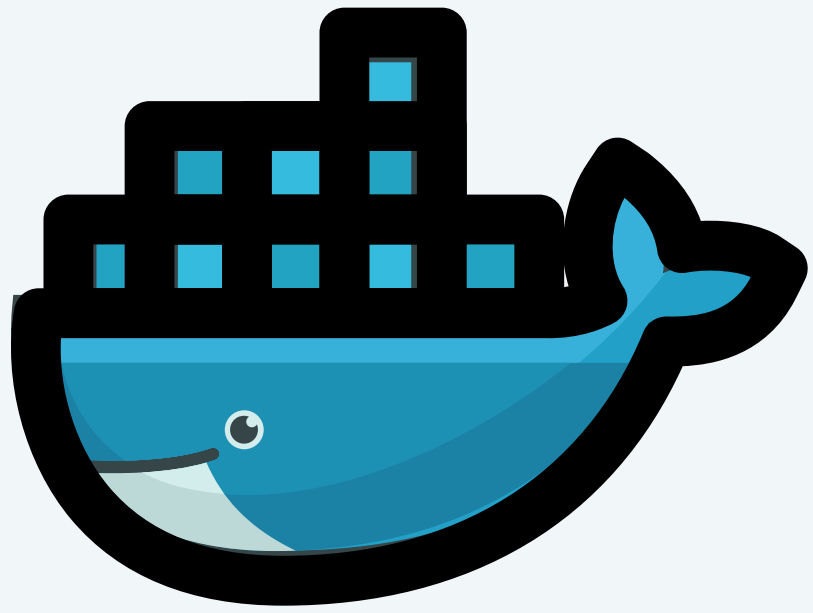
Los contenedores fallidos pueden ser fácilmente reemplazados.



PABLO DEL ÁLAMO

Ejemplo: Reinicio de Contenedor

Si un microservicio falla, Kubernetes puede reiniciar solo ese contenedor, evitando un caos mayor.



PABLO DEL ÁLAMO

Conclusión

La gestión de errores en microservicios, se trata de tener las herramientas adecuadas para controlar los errores rápidamente cuando ocurren, y evitar que estos hagan que el sistema se caiga por completo.

Con técnicas como retries, circuit breakers, timeouts y fallbacks, podemos minimizar el impacto de los fallos y asegurar que nuestras aplicaciones sigan operativas.

La clave está en la resiliencia y en ser proactivos con la monitorización y logging. Así, estarás siempre listo/a para manejar cualquier reto y mantener todo bajo control.

PABLO DEL ÁLAMO

¿Te ha resultado útil?

- Comparte esta guía con tu equipo o amigos desarrolladores.
- Guárdala para tenerla siempre a mano.
- ¡Dale un like o comenta si tienes preguntas!

PABLO DEL ÁLAMO