

MÓDULO 2: ARQUITECTURA DE SOFTWARE

ESTILOS DE ARQUITECTURA MICROSERVICIOS

INTRODUCCIÓN A LA ARQUITECTURA DE SOFTWARE

BIG school

Estilos de arquitectura: Microservicios

Servicios pequeños, autónomos, desplegables por separado



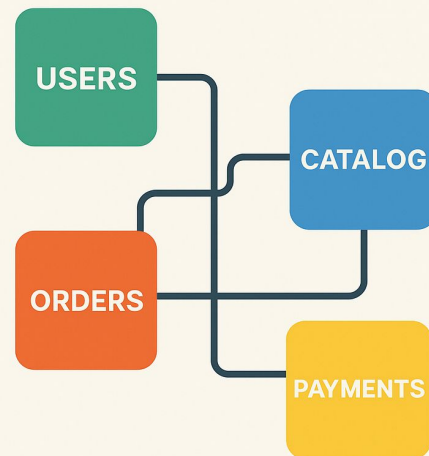
¿Qué son los microservicios?

Servicios pequeños e independientes

Una única responsabilidad de negocio

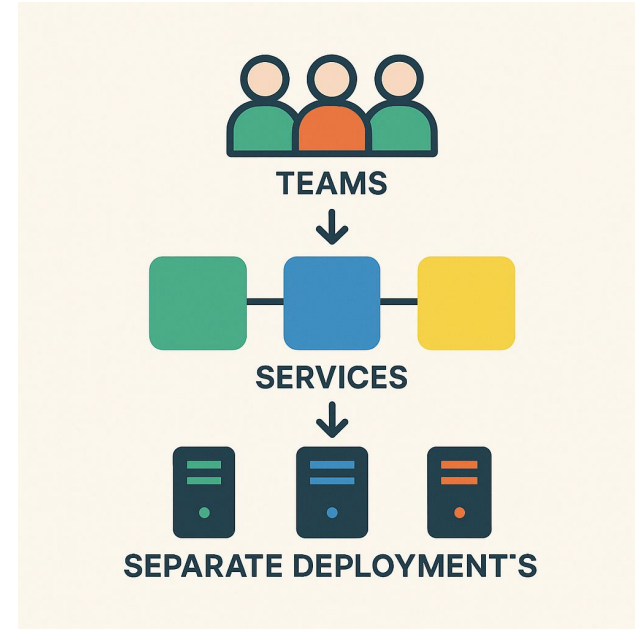
Comunicación vía **APIs o eventos**

Despliegue y escalado independientes



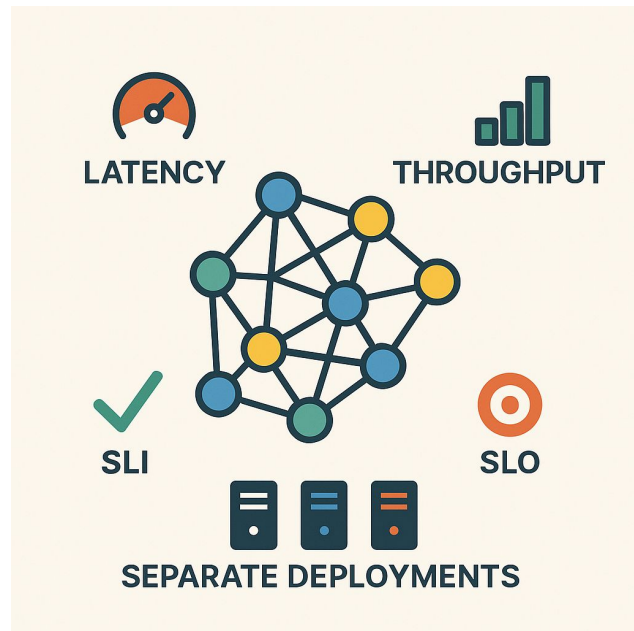
Beneficios

- ✓ Escalabilidad técnica (solo el servicio que lo necesita)
- ✓ Escalabilidad organizacional (equipos autónomos)
- ✓ Despliegues independientes
- ✓ Tolerancia a fallos



Retos y costes

- ❌ Complejidad operativa (infra, CI/CD)
- ❌ Observabilidad distribuida
- ❌ Consistencia eventual de datos
- ❌ Latencia de red
- ❌ Testing más costoso

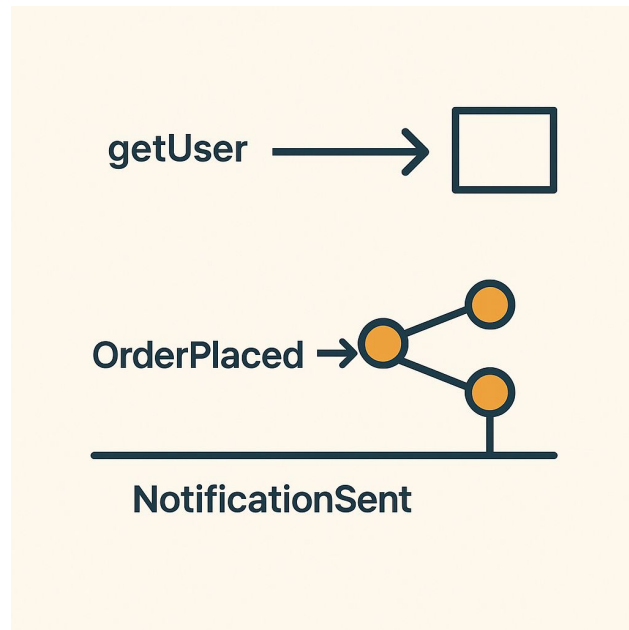


Comunicación

Síncrona → HTTP, gRPC

Asíncrona → Eventos, mensajería

Usar lo adecuado para cada caso



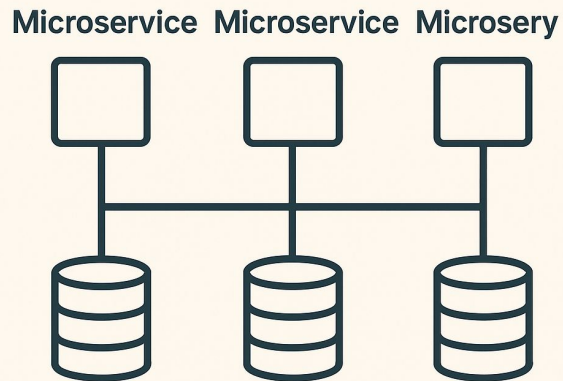
Datos

Cada servicio es **dueño de sus datos**

No hay joins entre servicios

APIs o eventos para compartir datos

Proyecciones locales cuando se necesita



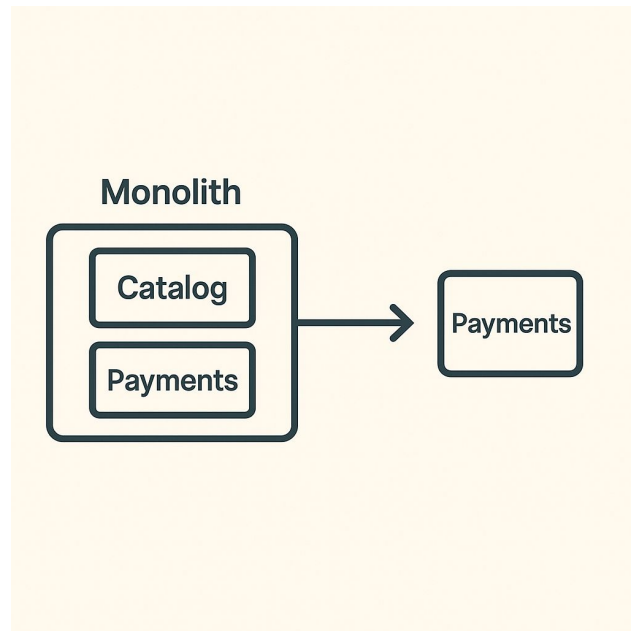
Evolución desde monolito

Empiezas con un monolito modular

Extraes módulo “caliente” como microservicio

Nueva BD + APIs/Eventos

Escalado y equipo propio



Buenas prácticas

Contratos claros (APIs, eventos versionados)

Automatización (CI/CD, tests, monitoreo)

Diseñar para fallos (timeouts, retries, circuit breakers)

Observabilidad obligatoria (logs, métricas, trazas)

Tamaño justo: pequeño, pero no demasiado



API



Event



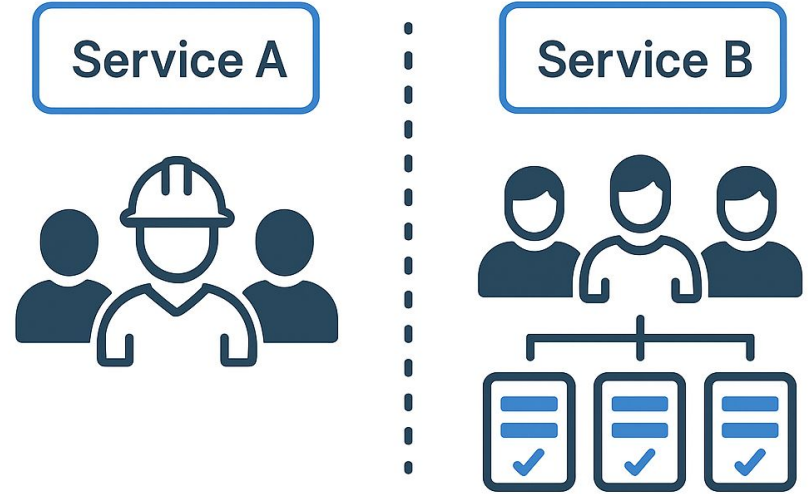
Discovery



Resilience

¿Cuándo convienen?


- ✓ Varios equipos paralelos
- ✓ Diferentes necesidades de escalado
- ✓ Despliegues independientes rápidos



¿Cuándo NO convienen?

- ✗ Equipo pequeño
- ✗ No tienes infraestructura madura
- ✗ Producto en validación inicial





*"Los microservicios te dan independencia y escalabilidad...
pero a cambio de complejidad."*

→ Próxima lección: **Arquitectura Hexagonal (Ports & Adapters)**