Análisis y Diseño de Aplicaciones 2

Facultad de Ingeniería

UT4 - TFU

Estudiantes

Agustín Schlechter

Diego Vázquez

Leandro Barral

Andrés Dandrau

Iván Lomando

Profesores

Ángel Mamberto

Esteban Roballo

Intro

Selección de 7 patrones exactamente: 2 de disponibilidad (CB, Health), 2 de rendimiento (QBLL, CQRS/MV), 2 de seguridad (Gatekeeper, Gateway Offloading + OIDC/Federated Identity), 1 de mod/despliegue (External Config Store). Todo alineado con la unidad.

Cada patrón se presenta con su diagrama de despliegue y diagrama de secuencia + explicación + medición empírica, y la Parte 2 queda cubierta con docker-compose (ya te lo di antes), curl/Postman, y scripts.

Repo

https://github.com/diegolagreca/UT4TFU

Disponibilidad (Circuit Breaker + Health Endpoint Monitoring)

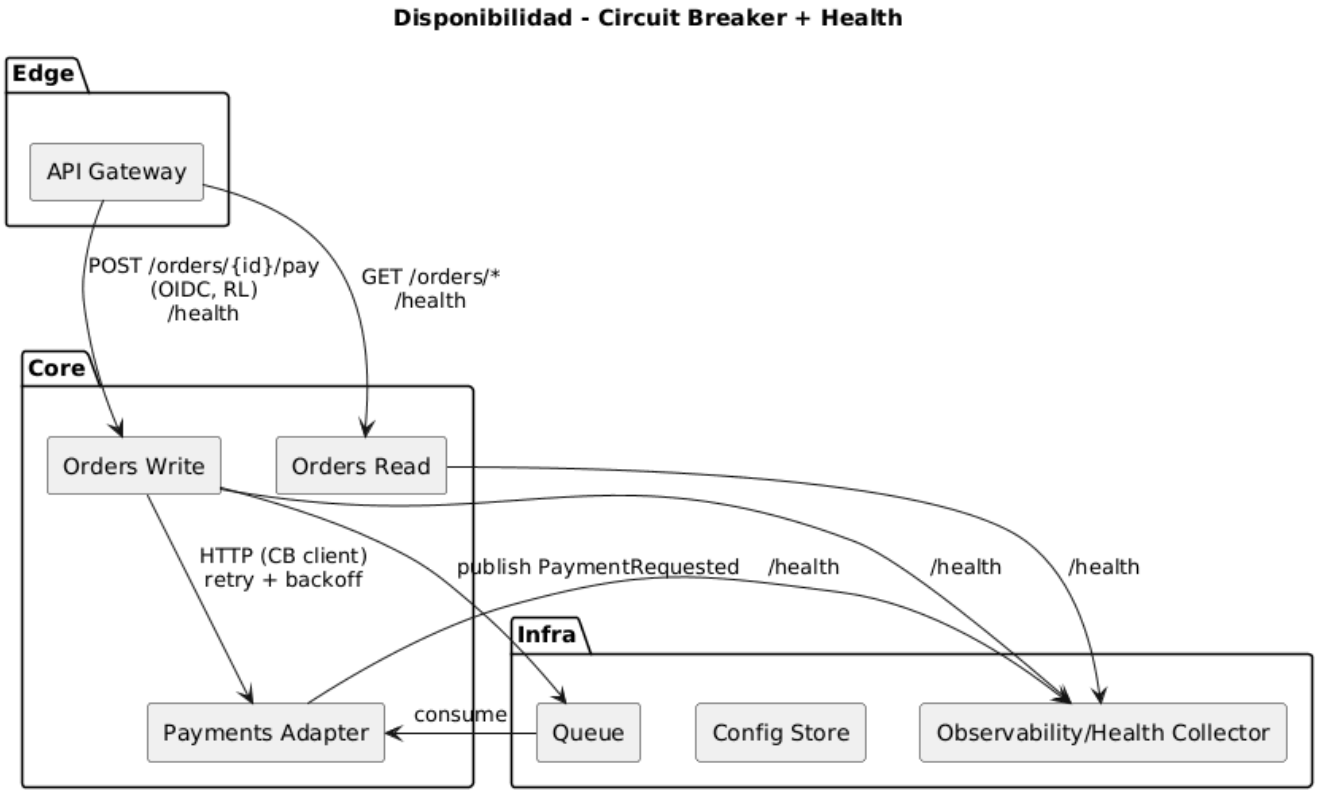
Componentes

- Circuit Breaker (CB) entre orders-write ↔ payments-adapter para evitar cascadas ante fallos; estados Closed → Open → Half-Open con fallback/degradación. (Defensa ante fallos y aislamiento del problema).
- Health Endpoint Monitoring en todos los servicios para que el orquestador/ELB y el gateway detecten y aíslen instancias malas.

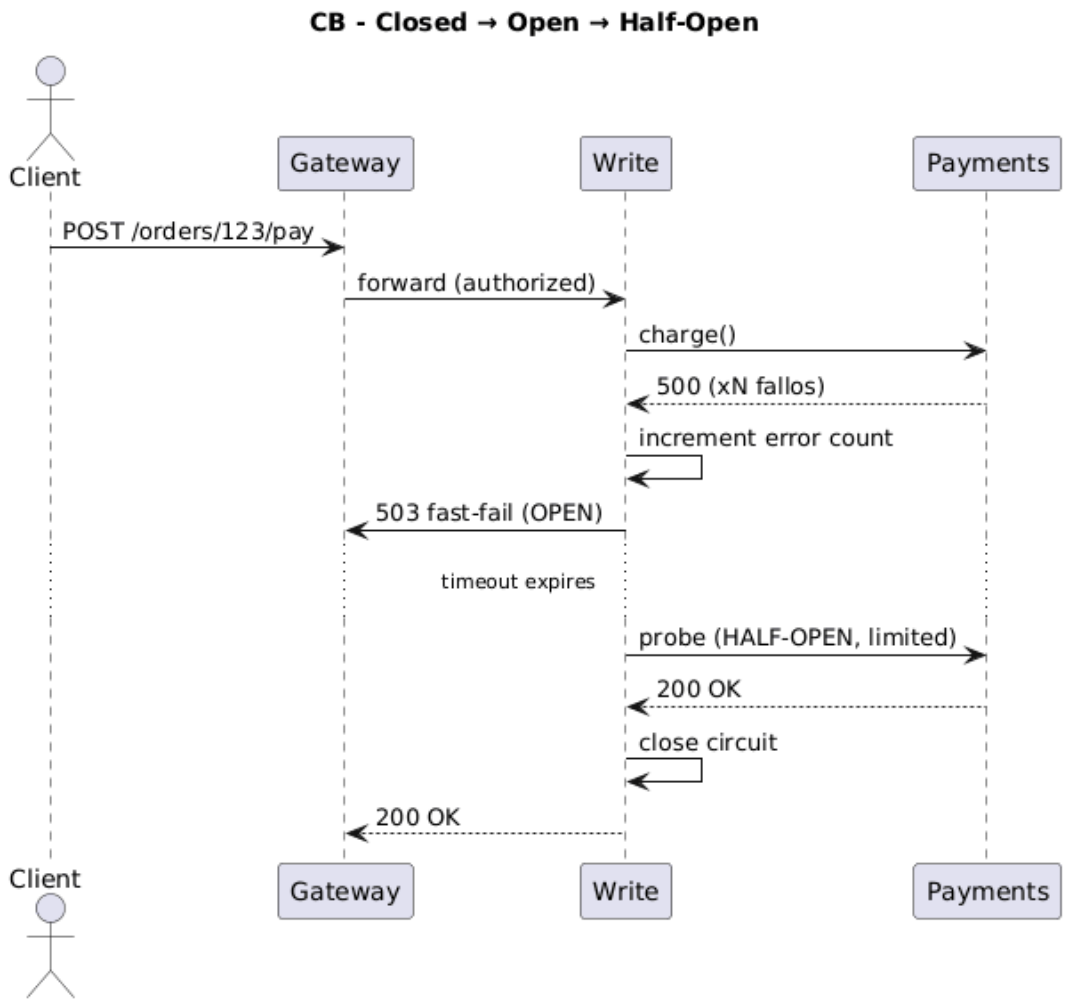
Implementación

- CB reduce el tiempo en que el usuario queda esperando un backend roto; “corta” y responde predecible, protegiendo recursos sanos.
- Health endpoints permiten detectar degradación/caídas y retirar nodos del pool.

UML CB + Health Monitor



Secuencia Circuit Breaker



Medición

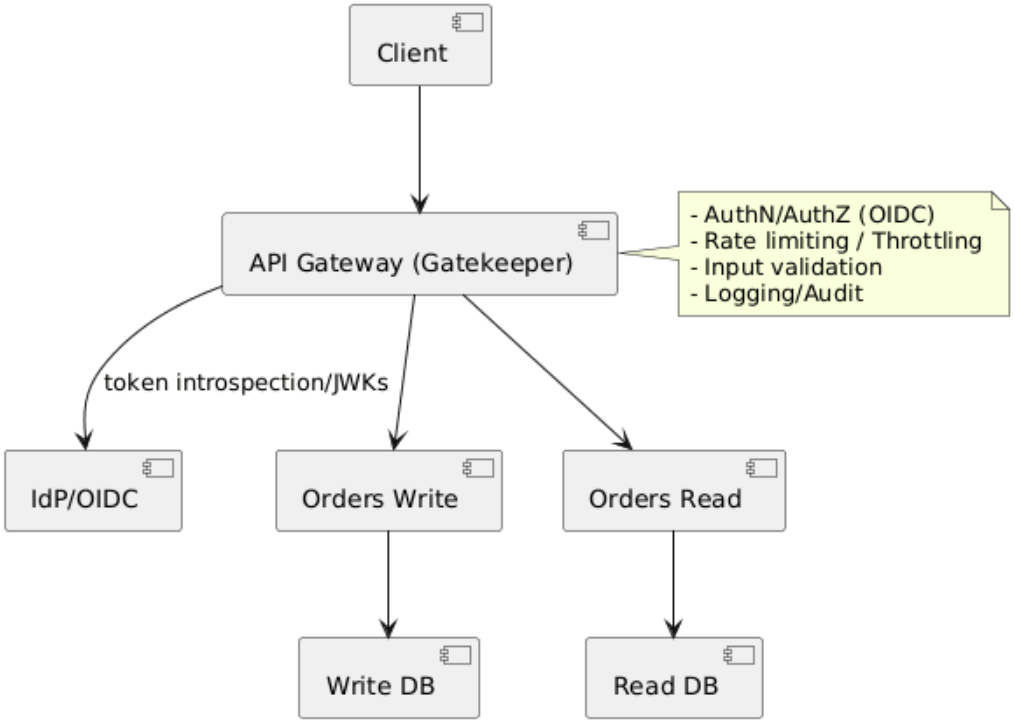
CB: simular 5xx sostenidos en payments-adapter y verificar:

1.1. tiempo hasta “open”

1.2. proporción de fast-fails vs intentos reales

1.3. éxito de sondas “half-open”.

Seguridad - Gatekeeper + OIDC Offloading



Rendimiento (Queue-Based Load Leveling + CQRS/Materialized View)

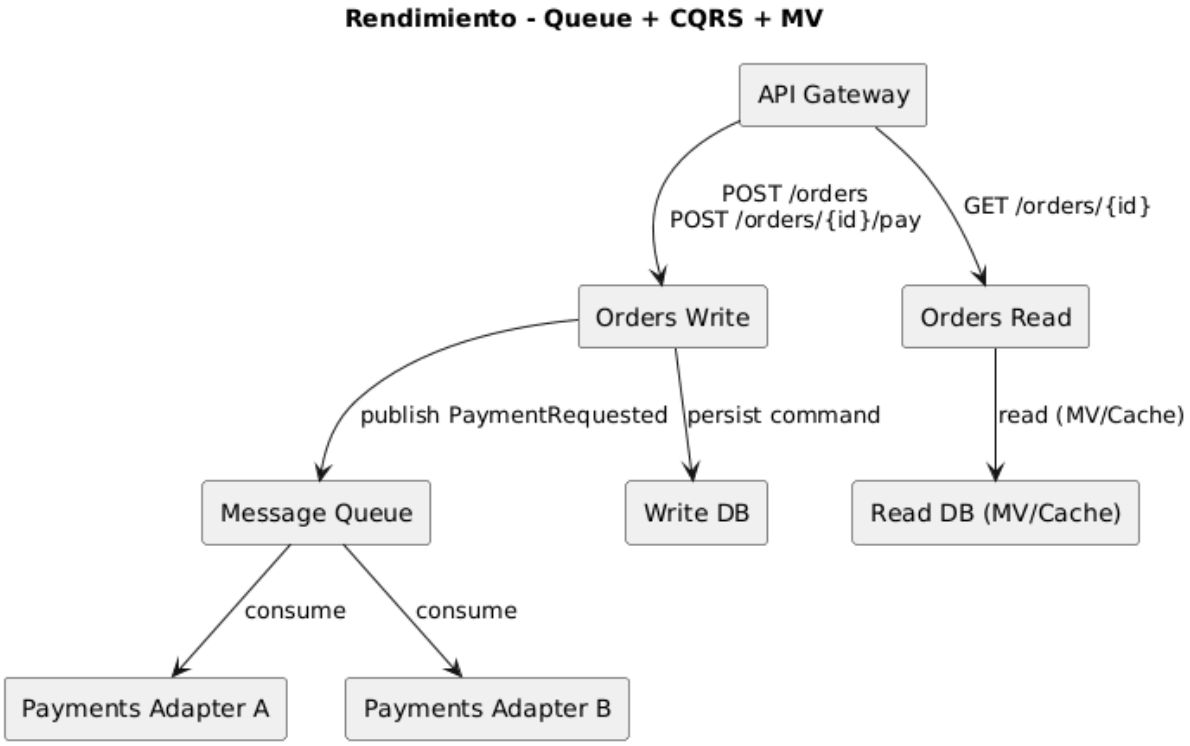
Componentes

- Queue-Based Load Leveling (QBLL) desacopla picos: orders-write encola pagos y payments-adapter los procesa estable; escalás con Competing Consumers.
- CQRS + Materialized View separa escritura/lectura y sirve lecturas rápidas desde vistas precalculadas (consistencia eventual).

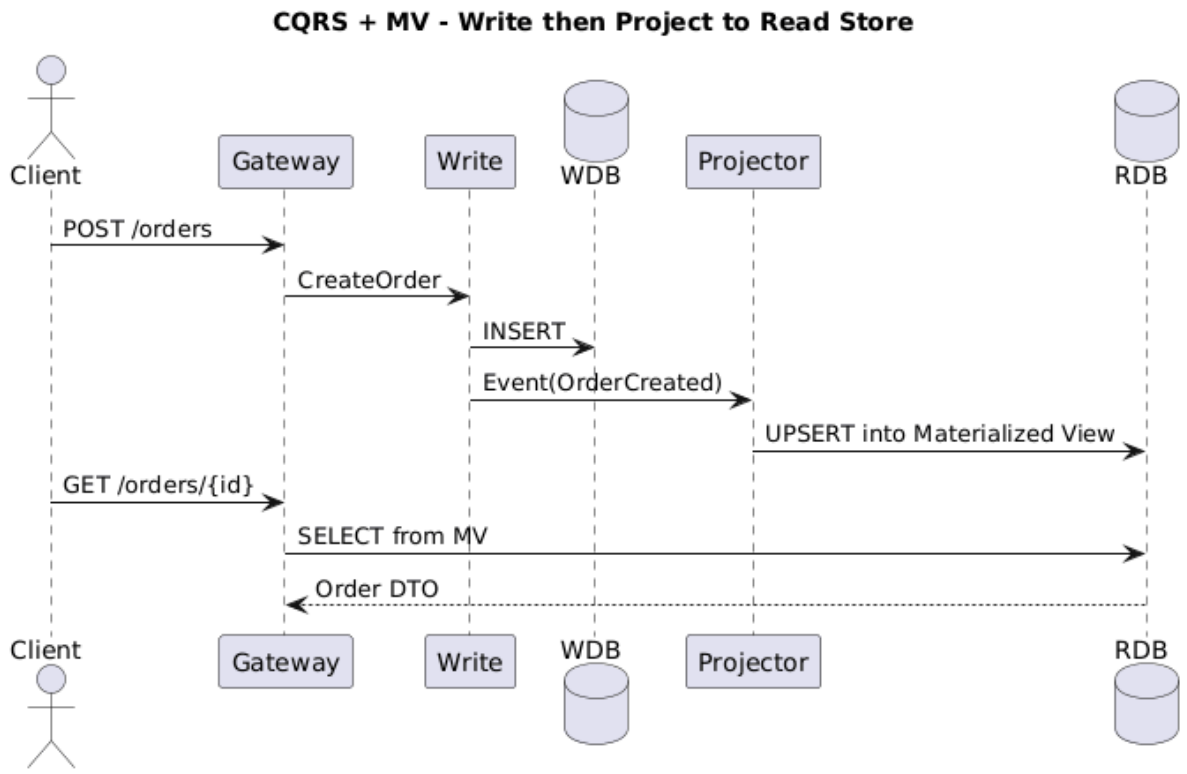
Implementación

- QBLL absorbe ráfagas, mantiene baja la latencia del endpoint productor y evita saturación de servicios críticos.
- CQRS/MV reduce sobrecarga de consultas complejas; P95/P99 bajan al servir desde vista optimizada.

UML QBLL + Competing Consumers + CQRS



Secuencia - Escritura y proyección a vista



Medición

- QBLL: inyectar 1000 pagos burst; medir latencia del POST (productor) y tiempo de vaciado de la cola; escalar consumidores y comparar throughput.
- CQRS/MV: comparr P95 de GET /orders/{id} contra lectura directa en DB; documentar consistencia.

Seguridad (Gatekeeper + Gateway Offloading + Federated Identity)

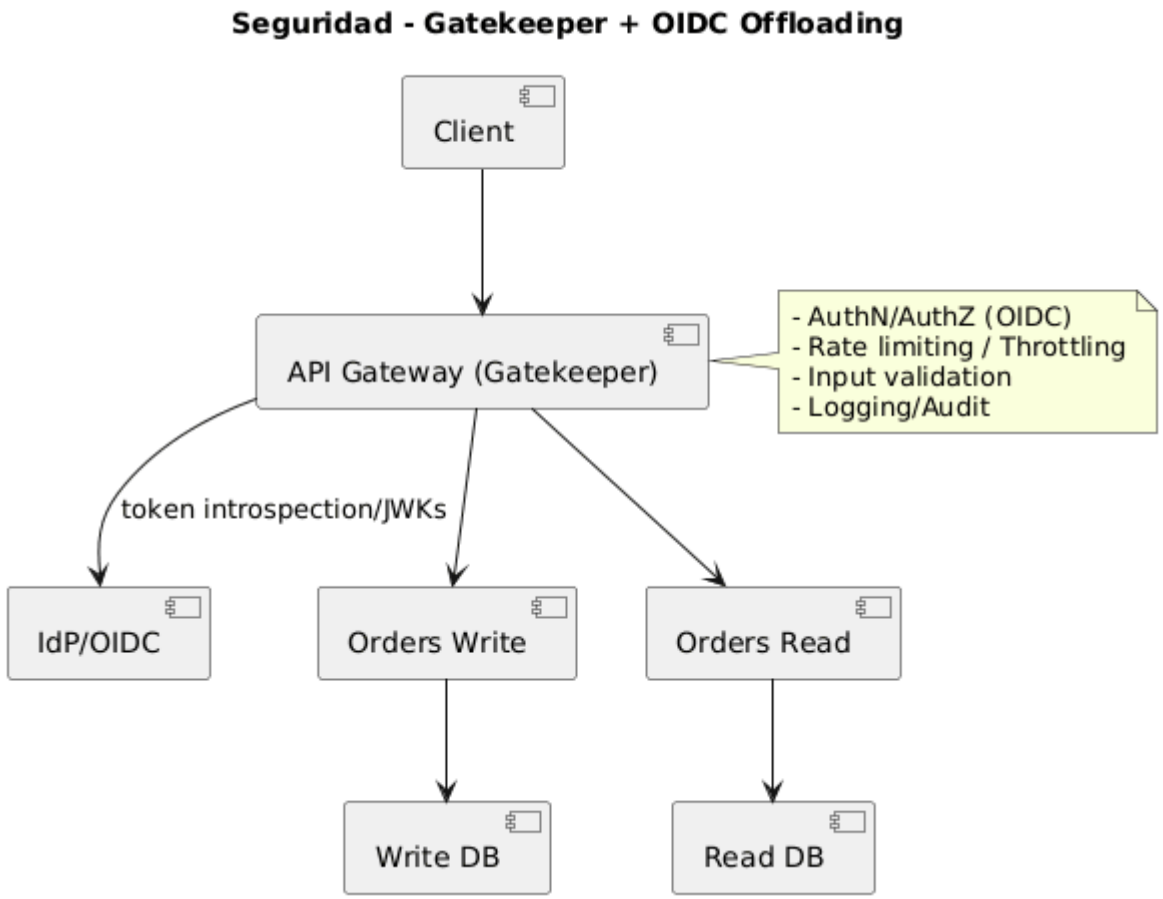
Componentes

- Gatekeeper (API Gateway) como punto único de entrada que valida, autentica, autoriza, rate-limitea, registra, y protege servicios internos.
- Gateway Offloading: el gateway descarga autenticación, validación de mensajes, cifrado, logging, etc., para que los microservicios focucen en negocio.
- Federated Identity (OIDC): el gateway confía en un IdP (p.ej., Keycloak) y aplica políticas por scopes/roles. (El material lista la familia de patrones y tácticas).

Implementación

- Centraliza políticas, reduce superficie de ataque y simplifica auditoría; riesgos: SPOF/latencia si no se diseña HA.
- Offloading estandariza controles (cifrado, validación de tokens, RL).

UML Gatekeeper + IdP



Medición

- AuthZ: sin token → 401, token inválido → 403, scope insuficiente → 403.
- RL: 100 req/10s → observar 429 a partir del umbral configurado en el gateway.

Facilidad de modificación/despliegue (External Configuration Store)

Componentes

- External Configuration Store: configuración centralizada versionable (flags de features, PAYMENT_MAX_RETRIES, límites de RL) consumida por los servicios en runtime. (Patrón cubierto en el set de despliegue/modificación; lo usamos para controlar comportamientos sin redeploy).

Implementación

- Cambios sin recompilar (feature toggles), rollback rápido, trazabilidad de cambios (auditoría).

UML Config central + hot-reload