Título

FASE 3: Choosing Monolith Decomposition pattern.

Estado

Aceptada

Fecha

2023-10-20

Contexto

- Ya tenemos el backend en cloud, nuestro siguiente hito será descomponer nuestra aplicación monolítica en microservicios, lo que dentro de las 7Rs llamamos (refactorización), para ello hemos analizamos diversos patrones de descomposición (Stangler Fig + Anticorruption Layer (ACL), Decorating Collaborator Pattern, Parallel Run, Branch by Abstraction). Recordemos que nuestro monolito tornará en una arquitectura totalmente serverless de microservicios.

Decisión

- De forma unánime el equipo decide que Stangler Fig el el patrón perfecto, nos permitirá ir reemplazando de forma gradual las partes monolíticas de la aplicación con microservicios que con el tiempo van estrangulando y reemplazando las funcionalidades del monolito original, detectamos que nuestros microservicios necesitarán llamar a funcionalidades dentro del monolito, por lo que estas deberán ser expuestas. Por lo que para evitar el cambio en el modelo de dominio o los datos usaremos el patrón anti corruption Layer ACL.

Consecuencias

Positivas:

- Permite una migración incremental y controlada, sin necesidad de reescribir toda la aplicación de una sola vez.

- Reduce el riesgo de introducir errores o interrumpir el funcionamiento de la aplicación existente.
- Facilita la adopción de nuevas tecnologías y prácticas, como la integración continua, el despliegue continuo o el desarrollo dirigido por pruebas.
- Mejora la escalabilidad, la fiabilidad y el rendimiento de la aplicación, al dividirla en componentes más pequeños y autónomos.

Negativas:

- Requiere una planificación cuidadosa y una coordinación entre los equipos de desarrollo, para evitar conflictos o inconsistencias entre los servicios.
- Implica un mayor esfuerzo de gestión y monitorización, al tener que lidiar con múltiples servicios, dependencias y puntos de integración.
- Puede aumentar la complejidad y la sobrecarga de la comunicación entre los servicios, al tener que definir e implementar protocolos, contratos e interfaces adecuados.
- Puede generar una duplicación temporal de código o funcionalidad, al tener que mantener tanto el código antiguo como el nuevo hasta que se complete la migración.

Compliance

- Nuestro cliente opera en Europa y Estados Unidos Por lo que ha de cumplir con los estándares GDPR e HIPAA)

Procedimiento

- Añadiremos la capa Proxy o Stranger Facade (Api Gateway) que se encargará de dirigir las peticiones al monolito o al microservicio según corresponda mediante pass-through
- Identificaremos los componentes a modernizar, pudiendo usar un diseño orientado a dominio (domain driven design (DDD).
- Creamos el nuevo microservicio que reemplace la funcionalidad del componente en cuestión.
- Configuraríamos el enrutamiento proxy (Api Gateway) para enviar las solicitudes hacia el nuevo microservicio.
- Vamos realizando la implementación de forma gradual.

- En nuestro caso los microservicios necesitan llamar a una funcionalidad dentro del monolito por lo que esta deberá ser expuesta. Para evitar el cambio en el modelo de dominio o los datos usaremos el patrón anti corruption Layer (ACL).

Notas

- Autor: Fran Díaz
- Version: 0.1
- Changelog:
    - 0.1: Versión Inicial propuesta.