Capítulo 5: Dividiendo aplicaciones monolíticas

Víctor Iranzo

25 de mayo de 2018

En este capítulo se explica como abordar la transformación de una aplicación monolítica de forma evolutiva hacia un sistema basado en microservicios. Las aplicaciones monolíticas tienden a crecer con el tiempo, por lo que se hacen frágiles y poco mantenibles al juntar muchas veces código no relacionado. Es por este motivo por el que un equipo pueda preferir no modificar una aplicación así, al menos no de forma descontrolada.

Parte I

Costuras: pasos para dividir lo monolítico

En el libro "Working Effectively with Legacy Code"se define una costura como una porción de código que se puede tratar de manera asilada sin alterar al resto del sistema. Las costuras son firmes candidatos a convertirse en futuros servicios.

El primer paso de nuestra refactorización será identificar las costuras. Muchos lenguajes de programación permiten la creación de espacios de nombres o paquetes. Si podemos, moveremos todo el código del contexto que hemos encontrado a un nuevo paquete mediante refactorizaciones del IDE.

Siguiendo este procedimiento, terminaremos viendo qué código se ha agrupado correctamente y qué código parece que no encaje en ningún paquete. El código sobrante puede estudiarse para ver si se puede agrupar como uno o varios paquetes o se puede añadir a la solución de otra forma.

Durante todo el proceso, el código debe representar una situación real, por lo que las interacciones y dependencias entre los paquetes será similar

a la existente en la realidad. También cabe mencionar que la transformación a microservicios no necesita realizarse de golpe, sino que se pueden ir transformando paquetes progresivamente para limitar el impacto de hacer algo mal. A la hora de elegir por qué paquete comenzar, podemos elegir el que supongamos que nos aportará mayor beneficio al separarlo del resto o el paquete que menos dependencias tenga con el resto.

Parte II

Razones para refactorizar aplicaciones monolíticas

- Tranquilidad de cambio: un cambio en un servicio se podrá hacer más rápido de lo que se hacia antes, además de poderse implementar y desplegar de forma autónoma.
- Organización de los equipos: diferentes equipos pueden encargarse de distintos microservicios sin que interfieran los cambios de unos con los de otros.
- Seguridad: la seguridad ahora puede aplicarse a un servicio concreto en lugar de a todo el sistema.
- Tecnológica: los servicios son más sencillos de refactorizar y cambiar a una nueva tecnología se puede hacer de forma progresiva y aislada en un único contexto.

Parte III

Refactorizaciones en bases de datos

La base de datos es la infraestructura donde más enredo de dependencias hay. El proceso a seguir para separar la persistencia de la aplicación en diferentes bases de datos consiste también en buscar costuras. Vamos a explicar algunos problemas que se pueden encontrar en este proceso.

1. **Claves ajenas**

2. **Datos estáticos compartidos**

3. **Datos mutables compartidos**

4. **Tablas compartidas**

Parte IV

Transacciones

Parte V

Interacción con grandes volúmenes de datos