

# Capítulo 2: La arquitectura evolutiva

Víctor Iranzo

17 de mayo de 2018

## 1. Visión y adaptación de la arquitectura

Con la facilidad de cambio que ofrecen las arquitecturas basadas en microservicios, el rol del arquitecto de software se ve afectado. Su papel principal será el de asegurar la calidad del software y tomar decisiones que ayuden a responder mejor a los cambios, porque el software ha de ser diseñado para ser flexible, adaptarse y evolucionar en función de los requisitos de los usuarios.

Los requisitos en la ingeniería del software cambian más rápidamente que los de otras profesiones. En lugar de centrarse en diseñar un producto final perfecto, el arquitecto debe crear un entorno donde el sistema correcto pueda emerger creciendo progresivamente a medida que se descubren nuevos requisitos.

Una de las responsabilidades del arquitecto de software es la de diseñar el sistema en la que tanto los usuarios como los desarrolladores se sientan cómodos. Para estos últimos, en la solución se debe promover la mantenibilidad, que en la ISO/IEC 25000, conocida como SQuaRE (System and Software Quality Requirements and Evaluation), se divide en las siguientes subcaracterísticas:

- Modularidad: capacidad de un sistema o programa de ordenador (compuesto de componentes discretos) que permite que un cambio en un componente tenga un impacto mínimo en los demás.
- Reusabilidad: capacidad de un activo que permite que sea utilizado en más de un sistema software o en la construcción de otros activos.
- Analizabilidad: facilidad con la que se puede evaluar el impacto de un determinado cambio sobre el resto del software, diagnosticar las

deficiencias o causas de fallos en el software, o identificar las partes a modificar.

- Capacidad para ser modificado: capacidad del producto que permite que sea modificado de forma efectiva y eficiente sin introducir defectos o degradar el desempeño.
- Capacidad para ser probado: facilidad con la que se pueden establecer criterios de prueba para un sistema o componente y con la que se pueden llevar a cabo las pruebas para determinar si se cumplen dichos criterios.

En definitiva, el arquitecto debe garantizar que la visión del sistema, que podríamos definir cómo las características y restricciones a alto nivel del sistema ha desarrollar, sea comprendida tanto por el equipo de desarrollo como por los clientes y evolucione en función de los requisitos de ambos grupos.

## 2. Colaboración entre microservicios

El arquitecto de software debe preocuparse más por como interaccionan los servicios entre ellos y no tanto en lo que ocurre dentro de los límites de cada uno de ellos. En organizaciones grandes, cada microservicio puede estar desarrollado por un equipo distinto y es el arquitecto quien debe hacer de puente entre ellos.

Como recordamos del capítulo anterior, una de las ventajas de las arquitecturas basadas en microservicios es la heterogeneidad tecnológica. Sin embargo, dejar plena libertad a cada equipo para elegir la tecnología del servicio que va a desarrollar puede traer problemas a la hora de integrarlo con el resto del sistema. En este sentido, una buena práctica sería:

- Establecer normas en aspectos clave como el protocolo de comunicación entre los servicios para facilitar el consumo entre ellos.
- Dejar mayor libertad al equipo de desarrollo en otros aspectos como la manera en que se implementa cada servicio de manera particular. De esta forma, cada equipo gana mayor relevancia a la hora de diseñar su propio microservicio y el arquitecto juega un papel menos técnico y más de supervisor y asesor.

### **3. Toma de decisiones e impacto de los cambios**

El diseño de software se basa continuamente en hacer balances entre diferentes alternativas. A la hora de tomar una decisión, cuanta más información se pueda recabar mejor. El arquitecto es quien mejor debe conocer los principios para tomar una decisión, que se pueden dividir en:

- **Objetivos estratégicos:** constituyen la dirección que desea tomar una organización a muy alto nivel, sin incluir aspectos a nivel técnico.