ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS
INGENIERÍA EN SOFTWARE

Aplicaciones Web Avanzadas

Ismael Rivas Hidalgo

Arquitectura Orientada a Servicios vs Microservicios

A grandes rasgos, estas arquitecturas tienen como base una idea similar: dividir las aplicaciones en componentes más pequeños y fáciles de administrar. Sin embargo, la mayor diferencia entre estas tecnologías radica en el propósito y el momento de su respectiva implementación. SOA (Arquitectura Orientada a Servicios) es un término puramente de software. Su objetivo es cumplir un esquema de diseño más eficiente en el desarrollo de aplicaciones, pero la cobertura se detiene allí [1]. De hecho, este enfoque no afecta a la infraestructura, sin embargo, hoy sabemos que las aplicaciones se han vuelto tan complejas que ocupan incluso hasta centros de datos completos. Por otro lado, la idea de los microservicios es extender la implementación de estas unidades más pequeñas de aislamiento a la infraestructura, así como al diseño de aplicaciones.

La tecnología SOA generalmente se refiere a una aplicación separada que comprende varios servicios internos. Este enfoque les otorga autonomía a los componentes y requiere únicamente que las aplicaciones compartan un esquema que hace que los servicios sean más portátiles y que el desarrollo de nuevas aplicaciones sea aún más fácil, sin embargo, debemos recordar que una API moderna hace lo mismo en gran medida al instalar una capa de servicios dedicados para encargarse de la lógica.

Los microservicios son tanto una arquitectura de infraestructura como una arquitectura de software. De hecho, se podría suponer que el código presente en el servicio se desarrolla de acuerdo con principios similares a SOA. Sin embargo, en el caso de los microservicios, la implicación es que la infraestructura en la que se ejecuta cada servicio es independiente de los demás. También se conserva su autonomía tanto a nivel de servidor como a nivel de código [2].

Este proceso también indica que el equilibrio de carga puede ocurrir servicio por servicio, lo que brinda a los equipos más oportunidades para optimizar el rendimiento en comparación con SOA. En otras palabras, aumentar la capacidad de un servicio adjunto es como lanzar de forma independiente nuevas instancias, lo que proporciona una escalabilidad acelerada. Esto también conduce a una resolución más rápida de los problemas, porque en un modelo de este tipo es muy probable que los servicios puedan dejar de funcionar sin que todo el sistema se caiga.

SOA es una práctica que el desarrollador ejecuta dentro de uno o más códigos de aplicación, mientras que los microservicios incluso dictan la estructura del equipo y los roles de TI: los equipos de desarrollo están aislados por los servicios que desarrollan. Esto también es cierto para las aplicaciones SOA, sin embargo, en el caso de SOA, se requiere que el desarrollador tenga un conocimiento profundo de cada servicio. Dado que los microservicios también pueden servir como mecanismos para entregar infraestructura, tienen un mayor impacto en el despliegue y en dicha infraestructura.

Los microservicios benefician a la infraestructura, pero también a las implementaciones. Estos son más rápidos porque implementan diferentes servicios individualmente. Basta con implementar solo aquellos que están actualizados y no toda la aplicación. Esta característica brinda una forma de continuidad en el aprovisionamiento y dentro de las implementaciones, porque las unidades más pequeñas se pueden implementar de forma independiente con menos riesgo.

Otro beneficio que rara vez se menciona es que los servicios, si están correctamente configurados y en contenedores, pueden persistir en cualquier nube sin dejar de ser parte de una aplicación cohesiva. Entonces, es posible tener un servicio que administre las conexiones de los usuarios y se ejecute en un centro de datos de Amazon en la costa este

de los Estados Unidos, y ese mismo servicio también se ejecute en un centro de datos de Azure en Europa. Esta flexibilidad es excepcional.

Pero también hay otros puntos importantes a considerar. La sobrecarga adicional que requieren los microservicios en comparación con la tecnología SOA es considerable. De hecho, los microservicios requieren un monitoreo más sólido, una automatización de lanzamiento más eficiente y equipos de gestión de TI y desarrollo más disciplinados [3].

Para aterrizar lo discutido en los párrafos anteriores, se ha rescatado un caso de estudio de la implementación de microservicios. Zalando SE es una empresa de comercio electrónico multinacional alemana con sede en Berlín, se enfoca principalmente en la venta de ropa y calzado [4]. El servicio tecnológico innovador que ofrecen aborda escenarios en los que los consumidores ya tienen una idea vaga de lo que están buscando. El objetivo del servicio es orientarlos en la dirección de los artículos de moda más específicos y relevantes lo más rápido posible.

Zalando define la arquitectura de microservicios de búsqueda en términos de capas lógicas. Cada capa se construye a partir de múltiples microservicios relacionados con requisitos similares. Las capas les permiten definir los límites operativos, aprovechando los problemas de entrega de servicios sin estado/con estado.

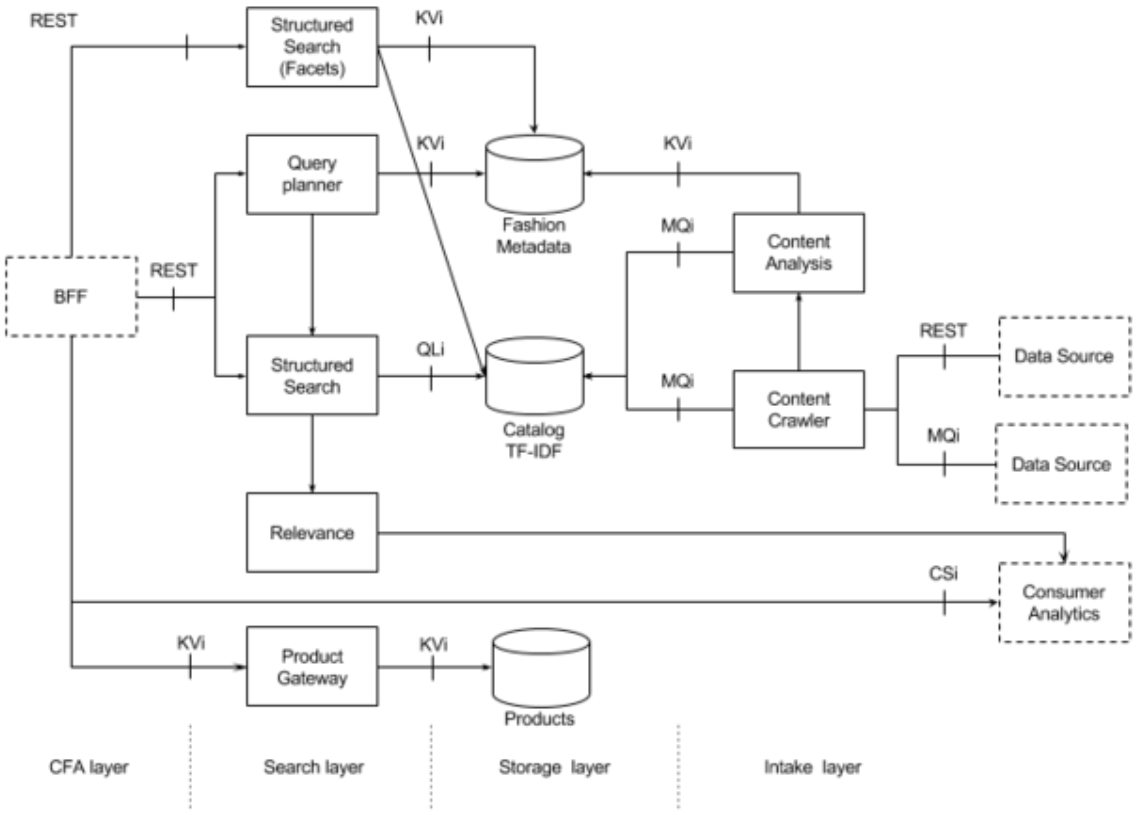

Ilustración 1. *Arquitectura de microservicios diseñada por Zalando.*

La capa de almacenamiento es un conjunto de componentes con estado que contienen datos de misión crítica. Se basa en tecnologías de código abierto disponibles en el mercado, siendo la parte central el sistema de recuperación de información TF-IDF . El uso de los servicios de valor agregado de AWS les ayuda a optimizar los procesos operativos y mejorar la durabilidad de los datos y la disponibilidad del servicio. La disponibilidad de la capa de almacenamiento

tiene un alto impacto en la experiencia del cliente y las ventas. Su estrategia de recuperación tiene como objetivo la alta disponibilidad (capacidad sobre el aprovisionamiento) y la recuperación automatizada de los nodos defectuosos.

La capa de entrada es una capa tradicional para extraer, transformar y cargar procesos; lo llaman canal de integración de contenido basado en eventos. Se utiliza para extraer contenido de una fuente heterogénea utilizando patrones de comunicación sincrónicos y asincrónicos. Por lo general, lee contenido de fuentes comerciales reales alojadas en la plataforma Zalando o extensiones de contenido impulsadas por una aplicación orientada al consumidor.

La capa de búsqueda se basa en microservicios orientados al consumidor que cumplen con el caso de uso de búsqueda. La capa es un microservicio puro y sin estado ajustado para la ejecución de un caso de uso "único". Por otro lado, una aplicación orientada al consumidor (CFA) es una granja de componentes backend para frontend (BFF) o proxies inversos que implementan una integración de casos de uso orientados al consumidor con infraestructura de búsqueda. Lo utilizan para descomponer los ciclos de vida de la evolución de la API de las aplicaciones a partir de la evolución del servicio.

En Zalando han descubierto desafíos arquitectónicos que contradicen ciertos estilos de desarrollo de microservicios, como las interfaces REST y el modelo de datos comunes. Aprendieron que los principios REST tradicionales basados en HTTP no se adaptan a todos los microservicios involucrados en su tecnología de búsqueda en moda o “Fashion Search”, como la han denominado. Han tomado la decisión de utilizar diferentes tecnologías API de acuerdo con sus necesidades, pero conservando su principio de diseño API First para que puedan escalar a medida que su departamento y negocio crece en alcance, evolucionando sus sistemas en paralelo.

Además, también han sabido garantizar la línea de base de interoperabilidad entre las aplicaciones orientadas al consumidor y los microservicios de la plataforma en ausencia de técnicas sólidas de negociación de contenido. Por lo tanto, han definido los principios del modelo de datos comunes que se utilizan en los microservicios dentro de la solución “Fashion Search”.

La evolución del desarrollo de software se basa en grados consecutivos de abstracción. El siguiente paso es abstraer la infraestructura para que deje de ser un obstáculo. Si el enfoque SOA puede considerarse un concepto antiguo, su espíritu sigue muy vivo. Sin embargo, la tecnología SOA se detiene en la aplicación y sabiendo que la complejidad de las aplicaciones aumenta, debe pensar más allá de este límite. Una estrategia de microservicios toma la idea de que las aplicaciones están formadas por muchos componentes pequeños que se excluyen mutuamente y la extiende a la infraestructura. En un modelo de microservicios, la coherencia funcional, los esquemas y los recursos de back-end son todos los elementos que componen la aplicación.

Bibliografía:

[1] Krafzig, Dirk, Karl Banke, and Dirk Slama. Enterprise SOA: service-oriented architecture best practices. Prentice Hall Professional, 2005.

[2] Ghofrani, Javad, and Daniel Lübke. "Challenges of Microservices Architecture: A Survey on the State of the Practice." ZEUS 2018 (2018): 1-8.

[3] Xiao, Zhongxiang, Inji Wijegunaratne, and Xinjian Qiang. "Reflections on SOA and Microservices." In 2016 4th International Conference on Enterprise Systems (ES), pp. 60-67. IEEE, 2016.

[4] A. Giamas, "From Monolith to Microservices, Zalando's Journey", InfoQ, 2016. [Online]. Available: https://www.infoq.com/news/2016/02/Monolith-Microservices-Zalando/. [Accessed: 02- Mar- 2022]