Microservices trade-offs

Introducción

Los microservicios son tanto un estilo de arquitectura como un modo de programar software. Los microservicios son pequeños elementos independientes, que se comunican a través de APIs bien definidas.

Cada función se denomina servicio y se puede diseñar e implementar de forma independiente. Esto permite que funcionen separados (y también, fallen por separado) sin afectar a los demás.

Hay muchas nuevas tecnologías que se aprovechan del aumento de la cultura del microservicio, Docker, un “contenedor de software”, es quizás el ejemplo más relevante. También otras plataformas similares como Nomad y Kubernetes.

Compensaciones

Escalabilidad

Los microservicios permiten que cada servicio se escale de forma independiente según la demanda de la funcionalidad de la aplicación que admite. Esto permite a los equipos satisfacer las necesidades de infraestructura, medir con precisión el costo de funcionalidad y disponibilidad si un servicio experimenta una mayor demanda.

Pero realmente en aplicaciones que necesiten escalabilidad, un buen diseño y desarrollo del software, unos pocos servidores básicos y un equilibrador de carga simple resolverían el problema de la escalabilidad sin falta de recurrir a los microservicios que sería más complejo.
Sólo en unos casos concretos compensaría usar microservicios para aumentar la escalabilidad.

Simplicidad

La complejidad esencial no desaparece solo porque el código se organice en módulos. Para las aplicaciones empresariales, la complejidad esencial se define en su mayor parte por los requisitos no funcionales que la solución necesita implementar. Esta complejidad no desaparece eligiendo un estilo arquitectónico diferente.

Con el uso de microservicios dividimos el código en pequeños métodos más simples y fáciles de entender. En el caso de que tuvieramos un programa de pocas líneas de código, sí nos sería útil organizarlo en servicios con métodos de un par de líneas, pero en el caso

de que tuvieramos una aplicación con miles de líneas de código, el hecho de dividir estas en cientos de métodos es incluso más difícil de entender.
En resumen, a mayor sea la aplicación, el uso de microservicios aumentará la complejidad estructural.

Reusabilidad

La reutilización significa un alto acoplamiento. Si el módulo falla, provocaría un error en cascada ya que aquellas partes donde fue reutilizado también fallarían.
Por lo que reutilizar microservicios significa paralizar deliberadamente la disponibilidad.
Además de que la reutilización rara vez se utiliza en un contexto de servicio.

Autonomía

La autonomía comienza en el nivel organizacional. En primer lugar, debe organizarse de manera que los equipos puedan tomar decisiones sin tener que coordinarse con personas ajenas al equipo. Para ello se debe establecer una autoridad de decisión descentralizada. Necesita organizar su fuerza laboral de manera que la toma de decisiones descentralizada sea posible y proporcione una clara ventaja sobre la toma de decisiones centralizada.

Los microservicios son un medio posible para respaldar la autonomía sin embargo, los microservicios por sí solos no permiten ni mejoran la autonomía. La autonomía debe establecerse a nivel organizativo, funcional y de gobierno. Los microservicios hacen las cosas más complejas.

Mejor diseño

Un desarrollo monolítico no tiene porqué estar mal estructurado, puede tener un buen diseño y una estructura mantenible. Por lo que hacer uso de un diseño con microservicios no quiere decir que vaya a ser mejor que uno monolítico.
Una estructura con microservicios es más difícil de desarrollar por lo general, por lo que si no somos capaces de desarrollar un sistema monolítico con una estructura correcta, tampoco lo seríamos con microservicios.

En el caso de que tuviéramos la capacidad de diseñar una buena modularización funcional entonces sí que sería mejor la opción de los microservicios.

Tecnología

Se cree que los microservicios facilitan la exploración de nuevas tecnologías y el cambio a estas. Esto es parcialmente cierto, ya que es más fácil migrar una pequeña parte, como es un microservicio, a una nueva tecnología que migrar una aplicación completa.

Pero realmente el esfuerzo de migrar una aplicación completa que se compone de muchos microservicios será tan costoso como migrar un sistema monolito.
La principal ventaja es que la migración resulta más fácil ya que podemos migrar la aplicación servicio por servicio, por lo que podemos equilibrar mucho mejor la migración.

Distribución

Los microservicios usan un sistema distribuido para mejorar la modularidad. Pero el software distribuido tiene una serie de problemas, el primero es el rendimiento, el segundo es utilizar asincronía ya que es difícil de utilizar y de depurar. Otro problema es la confiabilidad, tu esperas que haga llamadas a funciones y en ocasiones pueden fallar. Se pueden mitigar estos problemas pero puede resultar muy costoso y puede llegar a complicar mucho las cosas.

Alternativas

"MODULITHS"

Un modulito es un monolito formado por módulos claramente definidos y aislados.
Es una buena alternativa si:

- -No necesita ir rápido
- -No tiene varios equipos
- -No tiene requisitos funcionales muy dispares

Ofrece la mayoría de las ventajas de los microservicios, como una mejor capacidad de mantenimiento y evolución, independencia del equipo y más, sin exhibir las complejidades de los microservicios en tiempo de ejecución. Un modulito sigue siendo un monolito de implementación, es decir, una de las arquitecturas de tiempo de ejecución más simples disponibles.
Un modulito requiere habilidades avanzadas de diseño funcional y un estricto rigor de implementación, dos propiedades que, lamentablemente, no se ven con mucha frecuencia en la mayoría de las empresas

“MICROLITHS”
Un microlito es básicamente un servicio que se diseña utilizando los principios de diseño de módulos independientes que evita llamadas entre módulos / servicios mientras se procesa una solicitud externa.
Es una buena alternativa si:

- -No necesita ir rápido
- -No tiene varios equipos
- -Y tiene requisitos funcionales muy dispares
- -No está dispuesto a pagar el precio de los microservicios

Los microlitos son fáciles de manejar en tiempo de ejecución y facilitan los cambios tecnológicos.
Los microlitos siguen siendo servicios pero están mucho más cerca de lo que el desarrollador de software empresarial típico está acostumbrado y puede manejar de manera segura.

PARTICIPANTES

Miguel Fernández Álvarez UO269736
Alba Aparicio Pérez UO269926
Alex Caso Díaz UO269855

Bibliografía

https://www.redhat.com/es/topics/microservices
https://medium.com/@goodrebels/microservicios-ventajas-y-contras-de-la-arquitectura-descentralizada-a3b7fc814422
https://martinfowler.com/articles/microservice-trade-offs.html
https://www.ufried.com/blog/microservices_fallacy_1/
https://www.ufried.com/blog/microservices_fallacy_2_scalability/
https://www.ufried.com/blog/microservices_fallacy_3_simplicity/
https://www.ufried.com/blog/microservices_fallacy_4_reusability_autonomy/
https://www.ufried.com/blog/microservices_fallacy_5_design/
https://www.ufried.com/blog/microservices_fallacy_6_technology/
https://www.ufried.com/blog/microservices_fallacy_7_actual_reasons/
https://www.ufried.com/blog/microservices_fallacy_8_choices/
https://www.ufried.com/blog/microservices_fallacy_9_moduliths/
https://www.ufried.com/blog/microservices_fallacy_10_microliths/