uc3m | Universidad Carlos III de Madrid

Grado Ingeniería de Sistemas Audiovisuales

2018-2019

Trabajo Fin de Grado

“Diseño e implementación de un microservicio con Spring”

Jesús Rienda Iáñez

Tutor/es

Carmen Pelaez Moreno

Leganés, 2019

RESUMEN

Palabras clave:

DEDICATORIA

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. Planteamiento del problema

Tenemos la necesidad almacenar una serie de datos en una base da datos para posteriormente consultarlos, actualizarlos o crear nuevos registros. Por lo que necesitamos crear un servicio que cuando llamemos nos devuelva los datos almacenados con un tratamiento especifico y un formato definido. Necesitamos que sea sencillo y simple para el cliente que va a consumir dicho servicio. Este servicio tendrá que tener una alta disponibilidad y escalarse cuando sea necesario para siempre tener unos tiempos de respuesta bajos.

1.2. Solución propuesta

Para poder resolver nuestro problema usaremos un protocolo REST para la transferencia de datos entre el servicio y el cliente. El servicio web sera un microservicio desarrollado con Spring Boot. En cuanto a la base de datos utilizaremos una base de datos PostgreSQL.

1.3. Justificacion de la solucion

Para poder decidirnos sobre que arquitectura usar para el desarrollo de nuestra aplicación hemos tenido que investigar sobre las tecnologías existentes que pueden ayudarnos a solucionar el problema, las mas utilizadas son las arquitecturas monolíticas como SOA() y últimamente se usa mucho la arquitectura de microservicios. Como necesitamos un alto grado de escalabilidad los microservicios gestionan mejor esta propiedad ya que directamente en el contenedor donde se despliegan cuenta con un balanceador de carga el cual distribuye el trafico a la maquina que mas recursos tenga disponible. Para poder llegar a la decision de usar el protocolo REST hemos tenido que

1.4. Estado del arte

1.5. ¿Que es un Microservicio?

La arquitectura de microservicios pretende dividir una aplicación compleja en pequeños servicios que solo realicen una función especifica y se comuniquen entre ellos para formar la aplicación final.

Cada microservico es una aplicación pequeña totalmente independiente de desarrollar

frente al conjunto. Este servicio tiene responsabilidad única, es decir, un único requisito funcional. Ya que este es pequeño, si falla algo es fácil encontrar el error y subsanarlo.

Ademas al tener un servidor embebido es autoejecutable en cualquier contenedor de aplicaciones, por ejemplo Kubernetes, lo que facilita enormemente su despliegue.

Cada microservicio puede desarrollarse en un lenguaje de programación diferente, gracias a que las peticiones a estos servicios suele ser mediante protocolos ligeros, como HTTP, que son universales a todos los lenguajes de programación. También es fácilmente escalable, se le asignan mas recursos al microservicio que los necesite.

Los expertos en la materia no se ponen de acuerdo en el tamaño que debe tener un microservicio. Sam Newman hace referencia en su libro "Building Microservices"(cita) al experto en microservicios Jon Eaves que el tiempo máximo para desarrollar un microservicio una persona serian dos semanas. Aunque esto es muy relativo, ya que en una empresa grande la funcionalidad que debe realizar cada microservicio no tiene porque ser pequeña, por lo que llevaría mas tiempo.

1.6. Estado del arte

Actualmente la arquitectura basada en microservicios es una realidad y las grandes empresas tales como Netflix, Facebook, Amazon, PayPal se han replanteado la arquitectura que anteriormente usaban para decantarse por los microservicios debido a sus ventajas.

Rajest RV en su libro "Spring Microservices"[1] nos muestra un gráfico en el que se ve como avanza el desarrollo de aplicaciones tradicionales frente a microservicios. "Los microservicios prometen más agilidad, velocidad de entrega y escala. En comparación con las aplicaciones monolíticas tradicionales."

Los microservicios han evolucionado tan rápido gracias al desarrollo de nuevas tecnologías. Por ejemplo la creación de bases de datos NOSQL no relacionales, la aparición de docker contenedores de aplicaciones ligeros y portables que facilitan enormemente los despliegues y el comienzo del desarrollo de aplicaciones en la nube. Todo ello unido ha desembocado en la creación de microservicios.

Existen muchas razones por las cuales hacer el cambio o directamente empezar a desarrollar con esta arquitectura. - Los microservicios son mas rápidos y baratos de desarrollar que las antiguas aplicaciones monolíticas. - Es más fácil remplazar una parte de la aplicación global que en los sistemas antiguos. - No necesitas un servidor tipo tomcat donde desplegar la aplicación, ya que cada microserivcio cuenta con uno embebido. - Se suelen desplegar en la nube dentro de contenedores dockers por lo que no es necesaria una maquina física(servidor) 24 horas funcionando. - Puedes desarrollar cada microservicio en un lenguaje de programación diferente, en función de las necesidades.

Una metáfora que suele usarse para explicar los microservicios es un panal de abeja.

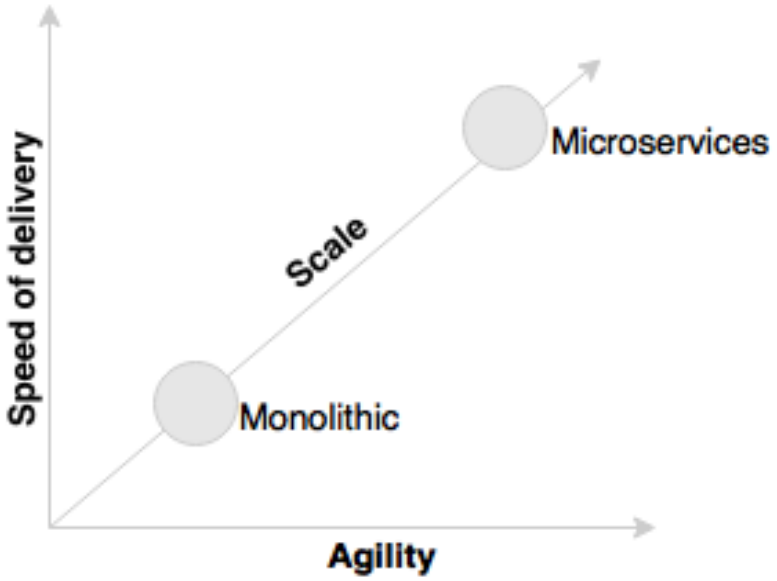

Fig. 1.1

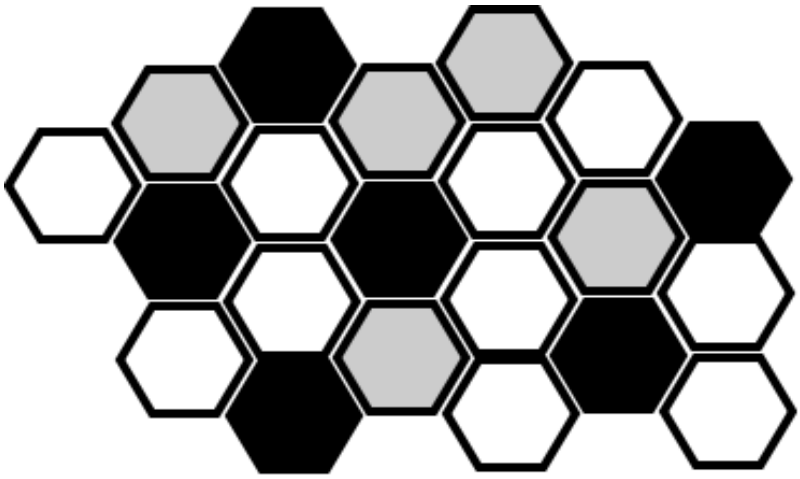

Fig. 1.2

Las abejas construyen el panal rellenado pequeñas celdas hexagonales, cada una independiente pero también integrada con las otras. Esto genera una estructura muy fuerte. Si una celda se daña no afecta a las demás, las abejas reconstruyen solo esa celda.

BIBLIOGRAFÍA

[1] R. RV, *Spring Microservices*. Packt Publishing, 2016. [En línea]. Disponible en: https://books.google.es/books?id=pwNwDQAAQBAJ.

[2] J. Thönes, "Microservices", *IEEE Software*, vol. 32, n.º 1, pp. 116-116, 2015. DOI: 10.1109/MS.2015.11.