UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Implementación de un API Gateway para una arquitectura de microservicios

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Alejandro Carrión Sanmartín

Tutor: Patricio Letelier Torres

Curso 2020-2021

Resumen

????

Palabras clave: ?????, ???

Resum

????

Paraules clau: ????, ?????????

Abstract

????

Key words: ?????, ?????

Índice general

Índice de figuras

CAPÍTULO 1
Introducción

El uso de arquitecturas de microservicios es una práctica ya establecida hoy en día en el mundo del desarrollo de *software*. Esta consiste en la construcción de servicios independientes, ejecutados en procesos diferentes, que se encargan de realizar funciones concretas y que trabajan de forma conjunta para lograr el objetivo u objetivos globales de la aplicación. Los beneficios que otorga este enfoque frente a la aproximación tradicional monolítica son muchos y muy variados. Algunos de ellos son:

- **Uso de diferentes tecnologías**. Cada microservicio puede estar construido con una tecnología diferente y puede utilizar diferentes mecanismos de persistencia.
- **Maniobrabilidad en los despliegues**. Ante cualquier cambio no es necesario desplegar la aplicación entera, solamente los microservicios implicados.
- **Escalabilidad y mantenibilidad**. Los microservicios, y la separación funcional que otorgan, facilitan el escalado de las diferentes partes de la aplicación de manera independiente. Lo mismo sucede con el mantenimiento, pudiendo crear equipos especializados.

Se puede obtener más información acerca de los microservicios en el artículo de James Lewis y Martin Fowler titulado "*Microservices*" [1].

Por contraparte, este tipo de arquitecturas aumentan la complejidad del desarrollo en algunos aspectos concretos como pueden ser el versionado de los microservicios o la coordinación de las comunicaciones entre ellos. Es aquí donde entra este trabajo, pues está enfocado a paliar otra de sus desventajas: la exposición de múltiples puntos de entrada al *backend*. Para ello se pretende realizar un *API Gateway*.

Un *API Gateway*, o *Gateway* para abreviar, se entiende como un componente que se interpone entre la interfaz de usuario (frontend) de una aplicación y los servicios en los que se basa (*backend*). Actúa como intermediario y es el único punto de entrada al mencionado *backend*. Además, es habitual que realice tareas de autenticación, monitorización u orquestación.

Como se aprecia en la definición anterior, el uso de un *Gateway* no queda restringido a arquitecturas de microservicios, también puede ser utilizado en monolíticas, si bien constituye una parte fundamental en las primeras. En estas, consigue ocultar los pequeños servicios que componen la aplicación y permite tener varias instancias de ellos en ejecución al mismo tiempo. La figura **??** muestra la comparativa de arquitecturas de microservicios cuando se utiliza un *Gateway* y cuando no.

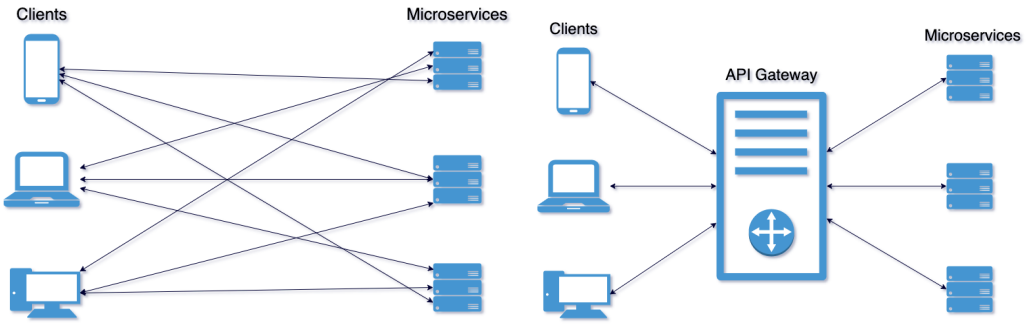

Figura 1.1: Arquitectura de microservicios básica frente a una con *API Gateway*. Imágenes de https://www.adictosaltrabajo.com/2020/05/27/introduccion-al-api-gateway-pattern.

1.1 Motivación

Este proyecto surge en el contexto de una práctica en empresa. El autor del mismo ha formado parte del equipo de *I+D+i* de la compañía *ADD Informática* durante el periodo de un año. *ADD* es una empresa española que se dedica a desarrollar y comercializar un *software* de gestión geriátrica llamado *ResiPlus* [2]. Actualmente cuenta con casi cien trabajadores, es líder del mercado nacional y se encuentra en proceso de expansión internacional.

La temática del trabajo es el desarrollo de un *API Gateway* para una aplicación con arquitectura de microservicios. Esta fue elegida debido a la relación que guarda con el desempeño del autor en la empresa comentada en el párrafo anterior. Por otro lado, el desarrollo a llevar a cabo le va a otorgar una visión más global de la aplicación sobre la que se trabaja así como aumentar el nivel de conocimiento acerca de la misma con la motivación de seguir contribuyendo al proyecto por mucho tiempo más. Por último, la tecnología a utilizar, *.NET Core*, es de su interés y aspira así a crecer como desarrollador de ese *framework*.

En cuanto a la motivación de la empresa, esta quiere dar la posibilidad de aprovechar, en cierta medida, el trabajo realizado y a realizar por el autor del documento dentro del marco de trabajo de la entidad, viéndose esta beneficiada también.

1.2 Objetivos

Este proyecto pretende resolver un problema real. Este consiste en otorgar a una aplicación con arquitectura de microservicios la capacidad de ocultarlos, así como permitir la ejecución simultánea de más de una instancia de los mismos. Cabe destacar que la aplicación en cuestión se encuentra todavía en fase de desarrollo y que el *API Gateway* que se va a realizar como solución es un punto clave en su arquitectura. A raíz de lo comentado se obtienen los siguientes objetivos sobre la aplicación en construcción:

- **Ocultar los microservicios** de forma que la interfaz de usuario no acceda directamente a estos por motivos de seguridad.
- Permitir la **instanciación múltiple** de los microservicios que, a su vez, tiene por objetivos:
  - Conseguir que la aplicación sea **tolerante a fallos** gracias a la posibilidad de tener un mismo microservicio desplegado en máquinas diferentes.

- **Aumentar la eficiencia** al poder crear o parar instancias dinámicamente según el tráfico que reciba la aplicación.

1.3 Estructura del documento

?????

CAPÍTULO 2
Contexto tecnológico

En la actualidad existen en el mercado muchas aplicaciones y servicios que resuelven problemas parecidos al planteado. Algunos de estos productos *software* son de pago, otros gratuitos e incluso algunos de código abierto. Se comentan los más relevantes a continuación.

2.1 *Amazon API Gateway*

Amazon API Gateway [3] es

2.2 *Kong*

Tal y como dice su página web, *Kong* [4] es el *API Gateway* más conocido. Se trata de una herramienta gratuita optimizada para microservicios, concebida para ser altamente escalable y que posee un gran número de *plugins* desarrollados por la propia compañía y por su comunidad, que es bastante grande al ser una aplicación tan popular. Además, actualmente es utilizado por grandes empresas como *Intel*, *Ferrari* o *Yahoo!*.

2.3 *Apigee*

Apigee [5] es

2.4 *Tyk*

Tyk [6] es

2.5 *Ocelot*

Ocelot [7] es

2.6 *YARP*

YARP [8] es

CAPÍTULO 3
Análisis del problema

?????

3.1 Especificación de requisitos

?????

3.2 Soluciones posibles

?????

3.3 Solución propuesta

?????

3.4 Plan de trabajo y metodología

?????

CAPÍTULO 4
Diseño de la solución

?????

4.1 Arquitectura del sistema

?????

4.2 Diseño detallado

?????

4.3 Tecnología utilizada

?????

4.4 Mantenimiento y gestión de versiones

?????

CAPÍTULO 5
Desarrollo de la solución

?????

CAPÍTULO 6
Pruebas

?????

CAPÍTULO 7
Conclusiones

?????

Bibliografía

[1] Artículo de J. Lewis y M. Fowler acerca de los microservicios. Disponible en: https://martinfowler.com/articles/microservices.html

[2] Página web de *ADD Informática* y su producto *ResiPlus*: https://addinformatica.com

[3] Documentación oficial de *Amazon API Gateway*: https://aws.amazon.com/es/api-gateway

[4] Documentación oficial de *Kong*: https://konghq.com/kong

[5] Documentación oficial de *Apigee*: https://cloud.google.com/apigee

[6] Documentación oficial de *Tyk*: https://tyk.io

[7] Documentación oficial de *Ocelot*: https://ocelot.readthedocs.io/en/latest

[8] Documentación oficial de *YARP*: https://microsoft.github.io/reverse-proxy