Patrones de diseño de microservicios: puerta de enlace API

por Andre Luis de Oliveira Dias MVB · 1 y 17 de diciembre · Zona de integración

Modernice sus arquitecturas de aplicaciones con microservicios y API con las mejores prácticas de esta serie gratuita de cumbres virtuales. Presentado en asociación con CA Technologies .

Patrones de diseño ... una herramienta valiosa en la vida de un arquitecto o un desarrollador inteligente. Tenemos cientos de ellos. Muchas formas de resolver problemas comunes, en muchas áreas. Desde el catálogo de patrones GoF (Gang of Four), hemos prestado la atención adecuada y hemos comenzado a utilizarlos en nuestros proyectos complejos, con gran éxito.

Los patrones de diseño están aquí para ayudarlo, sin importar cuál sea la complejidad del problema. El mundo de los microservicios no es una excepción.

Debido a muchos desafíos traídos por las complejidades del Estilo de Arquitectura de Microservicios, miramos hacia atrás y vimos muchos enfoques que tuvieron éxito hasta ahora. Ahora tenemos un consenso de Design Pattern en el formato de un catálogo para este tipo de soluciones arquitectónicas, que la comunidad actualiza y revisa constantemente.

En esta serie de artículos, resaltaremos los patrones de diseño más populares e importantes que debe considerar en la arquitectura de microservicios y qué herramientas o enfoques puede utilizar para aplicarlos de manera correcta y fácil.

En esta publicación, le proporcionaremos una introducción al tema general y elegiremos uno de los patrones de diseño más populares utilizados en las soluciones de microservicios: **el patrón API Gateway** .

Entonces empecemos.

Patrón # 1 - Pasarela API

Contexto

Los sistemas necesitan muchas piezas de información que se distribuyen en varios servicios dentro del catálogo de servicios. Además, muchos tipos de clientes consumen los servicios del sistema, agregando más complejidad a las soluciones finales cuando comenzamos a pensar en la evolución de esos servicios, respetando la comunicación y las soluciones de formato / tamaño de datos para cada uno de esos clientes.

De esta forma, los servicios deben ser capaces de "hablar" con cada uno de estos clientes (y ser fácilmente descubiertos) sin perder los requisitos no funcionales de rendimiento, mantenimiento, escalabilidad, seguridad y disponibilidad necesarios para respaldar estas soluciones distribuidas.

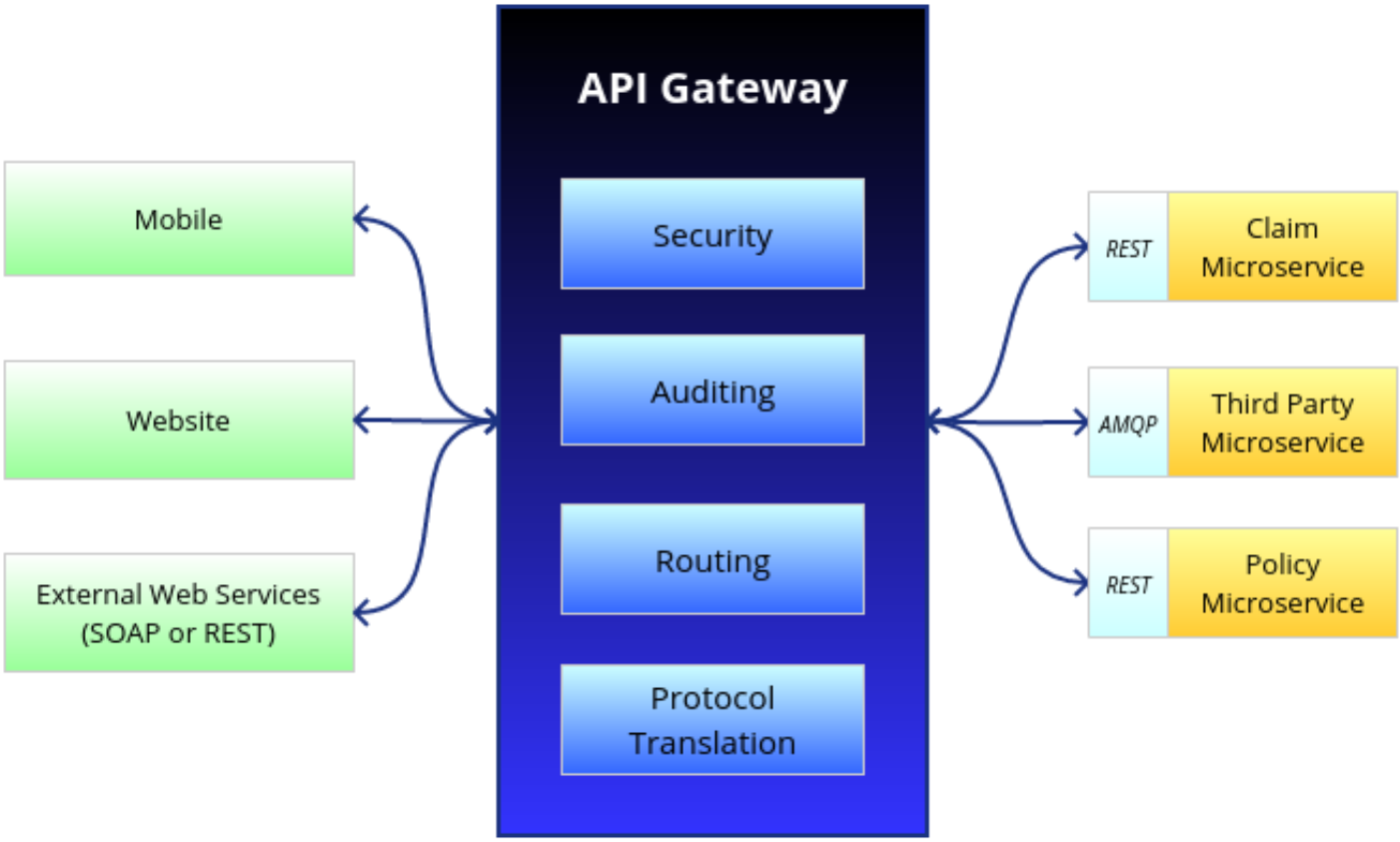
Para responder a la necesidad cada vez mayor de información y velocidad, se deben proporcionar cada vez más instancias de servicio para soportar la escalabilidad correcta, lo que agrega, por supuesto, más complejidad.

Entonces, el patrón API Gateway debería ser útil en este escenario ...

Objetivo principal

El objetivo de la API Gateway es representar un único punto de entrada para todos esos clientes y, al mismo tiempo, abstraer las complejidades de los problemas de comunicación, como las transacciones de protocolo y las conversiones de datos.

La siguiente imagen resalta el patrón API Gateway en la arquitectura de microservicios:



Beneficios directos

La puerta de enlace API es una solución independiente del protocolo que sirve a clientes distintos en

varios canales de comunicación. Expone cada API para cada consumidor, de acuerdo con la comunicación y los tipos de clientes, y adopta la seguridad una vez que es el principal punto de entrada. Resume cualquier estructura de refactorización o desglosamiento en los sistemas existentes (los llamados monolitos) para sus clientes, y abstrae la topología de microservicios subyacente y las tecnologías involucradas con los consumidores finales.

Inconvenientes

La puerta de enlace API también agrega una nueva capa de complejidad a la solución final de microservicios, tanto al implementar (o mantener) la puerta de enlace API con una tecnología / producto nuevo o existente y gobernar la evolución del contrato de la API, por lo que es importante para las soluciones distribuidas .

Como deberíamos esperar, agregar una nueva capa a la arquitectura general disminuye el tiempo de respuesta debido a los costos de conversación para solicitar a todos los microservicios involucrados en una transacción específica en un contexto empresarial particular.

Cómo implementarlo

Hoy las implementaciones disponibles más populares para realizar una solución de puerta de enlace API son:

- Kong (Fuente abierta)
- Spring Gateway (abierto procedente de Spring Cloud)
- Puerta de enlace API de Amazon
- Gestión de API de Microsoft Azure
- Gestión de la API de 3Scale de Red Hat

En las próximas publicaciones, exploraremos en detalle algunas de estas soluciones.

Resumen

Esencial para cada solución de microservicios, el patrón de puerta de enlace API ayuda a mejorar la mensajería de servicio y, al mismo tiempo, abstrae parte de la complejidad y los detalles que existen en cada llamada de microservicio. Al mismo tiempo, sirve como el principal punto de entrada para todo tipo de clientes.

Gracias por leer.

Me gusta este articulo? Leer más de DZone

¿Por qué los microservicios

Regule sus microservicios con

¿Por qué los microservicios necesitan una puerta de enlace API?

Regule sus microservicios con una puerta de enlace API: Parte I

Comunicación de microservicios: Zuul API Gateway

Gratis DZone Refcard
Primeros pasos con Spring Boot y Microservicios

Temas: MICROSERVICIOS, ARQUITECTURA DE MICROSERVICIOS, PATRONES DE INTEGRACIÓN, PUERTA DE ENLACE API, INTEGRACIÓN