Universidad Autónoma de Chiapas
Facultad de contaduría y administración C-I

Carrera:
Lic. En ing en desarrollo y tecnologías de software.

Materia:
Taller de desarrollo 4.

Catedrático:
Mtro. Luis Gutiérrez Alfaro.

Nombre del alumno:
González Aguilar Eduardo - A211154

Semestre: 6. **Grupo:** M.

Nombre de la actividad:
Arquitectura orientada a servicios.

Fecha de entrega:
19/08/2023.

Arquitectura orientada a servicios.

Arquitectura de microservicios:

La arquitectura de microservicios es un método de desarrollo de aplicaciones software que funciona como un conjunto de pequeños servicios que se ejecutan de manera independiente y autónoma, proporcionando una funcionalidad de negocio completa**.** En ella, cada microservicio es un código que puede estar en un lenguaje de programación diferente, y que desempeña una función específica**.** Los microservicios se comunican entre sí a través de APIS, y cuentan con sistemas de almacenamiento propios**,** lo que evita la sobrecarga y caída de la aplicación.

Características:

- Los microservicios son pequeños e independientes, y están acoplados de forma precisa. Un único equipo reducido de programadores puede escribir y mantener un servicio.

- Cada servicio es un código base independiente que puede administrase por un equipo de desarrollo pequeño.

- Los servicios pueden implementarse de manera independiente. Un equipo puede actualizar un servicio existente sin tener que volver a generar e implementar toda la aplicación.

- Los servicios son los responsables de conservar sus propios datos o estado externo. Esto difiere del modelo tradicional, donde una capa de datos independiente controla la persistencia de los datos.

- Los servicios se comunican entre si mediante API bien definidas. Los detalles de la implementación interna de cada servicio se ocultan frente a otros servicios.

- Admite la programación poliglota. No es necesario que los servicios compartan la misma pila de tecnología, las bibliotecas o los marcos.

Beneficios:

- **Agilidad:** Es más fácil administrar las correcciones de errores y las versiones de características. Además, puede actualizar un servicio sin volver a implementar toda la aplicación y revertir una actualización si algo va mal.

- **Equipos pequeños y centrados:** Un microservicio debe ser lo suficientemente pequeño como para que un solo equipo de características lo pueda compilar, probar e implementar

- **Base de código pequeña:** La arquitectura de microservicios minimiza las dependencias y resulta más fácil agregar nuevas características

- **Mezcla de tecnologías:** Los equipos pueden elegir la tecnología que mejor se adapte al servicio de una combinación de pilas de tecnología, según corresponda.

- **Aislamiento de errores:** Diseña una solución para que los microservicios se comuniquen entre si mediante patrones de mensajería.

- **Escalabilidad:** Los servicios se pueden escalar de forma independiente, lo que permite escalar horizontalmente los subsistemas que requieren mas recursos, sin tener que escalar horizontalmente la aplicación.

Técnicas de integración:

Las técnicas de integración arquitectura orientada a microservicios son formas de conectar y coordinar los servicios que componen una aplicación distribuida. Algunas de las técnicas más comunes son:

- **Coreografía**: Los servicios se comunican entre sí mediante eventos, sin un coordinador central. Cada servicio es responsable de suscribirse, publicar y procesar los eventos que le interesan.
- **Orquestación**: Un servicio especializado actúa como un coordinador central que dirige el flujo de la lógica de negocio y llama a otros servicios según sea necesario[1].
- **API Gateway**: Un punto de entrada único que expone una API para los clientes y enruta las solicitudes a los servicios adecuados. El API Gateway puede implementar funciones adicionales como autenticación, autorización, balanceo de carga, almacenamiento en caché, etc[23].
- **Patrón de agregación**: Un servicio compuesto que invoca varios servicios y combina sus resultados en una sola respuesta

Balanceo de carga:

Consiste en distribuir uniformemente el tráfico de red o de aplicaciones entre un grupo de servidores o endpoints. El objetivo esencial es garantizar que todos los recursos puedan atender las solicitudes. Mejora el funcionamiento resistente y eficiente de las aplicaciones web. Dirige de forma inteligente las solicitudes de los clientes a través de varios servidores y garantiza una utilización óptima de los recursos, el rendimiento de la aplicación y un tiempo de actividad fiable.

Despliegue:

La arquitectura de microservicios puede dividirse en varias operaciones de componentes separados. Esto permite el despliegue, la modificación y el redespliegue de servicios por

separado sin poner en peligro la estructura del sistema. En lugar de volver a desplegar aplicaciones completas, sólo habría que modificar un servicio específico de esta manera. Sin embargo, esta estrategia tiene inconvenientes, como las costosas llamadas remotas en lugar de las realizadas en el proceso y las mayores complicaciones a la hora de distribuir las tareas entre los elementos.

Arquitectura monolítica vs arquitectura de microservicios:

- **Estructura de la aplicación:** En las arquitecturas monolíticas, todos los componentes se agrupan en una unidad indivisible, mientras que las arquitecturas de microservicios organizan los componentes en servicios independientes más pequeños que se centran en capacidades comerciales específicas.

- **Desarrollo e implementación:** Las aplicaciones monolíticas son más sencillas de desarrollar e implementar debido a la naturaleza singular de la arquitectura. Aun así, las aplicaciones de microservicios requieren más esfuerzo en el manejo de la implementación, orquestación y monitoreo de componentes individuales. A pesar de la complejidad adicional, las arquitecturas de microservicios brindan más flexibilidad y permiten la implementación independiente de componentes, lo que acelera el lanzamiento de funciones y reduce el riesgo de fallas.

- **Escalabilidad:** Las aplicaciones monolíticas a menudo enfrentan desafíos de escalado, ya que agregar recursos requiere escalar todo el monolito, lo que puede consumir muchos recursos y ser ineficiente. Por el contrario, las arquitecturas de microservicios permiten el escalado independiente de los servicios en función de sus requisitos específicos, lo que da como resultado una asignación de recursos eficiente y un rendimiento mejorado

- **Mantenibilidad:** las aplicaciones monolíticas pueden ser difíciles de mantener debido a la interdependencia de los componentes. La modificación de un componente puede tener efectos en cascada en toda la aplicación, lo que aumenta el riesgo de fallas y dificulta la realización de correcciones y actualizaciones rápidamente. Las arquitecturas de microservicios permiten una mejor capacidad de mantenimiento al permitir el desarrollo independiente y las actualizaciones de componentes con un impacto mínimo en otros servicios.

- **Pila de tecnología:** las aplicaciones monolíticas suelen tener una única pila de tecnología unificada, lo que puede limitar la flexibilidad a la hora de elegir las mejores herramientas para tareas específicas. Por otro lado, las arquitecturas de microservicios permiten diferentes pilas de tecnología dentro de cada servicio, lo que permite a los equipos elegir las herramientas más adecuadas para sus necesidades específicas

Diseños de arquitecturas de microservicios.

- ### Aplicación con el cliente.

  Un diseño de bus de servicio es una forma de implementar la comunicación entre microservicios que usan un intermediario de mensajes, como RabbitMQ o Azure Service Bus. Este patrón puede ayudar a desacoplar los microservicios y mejorar la escalabilidad y la confiabilidad

- ### Servicios:

Este estilo ofrece ventajas como la escalabilidad, la resiliencia, la independencia y la evolución rápida, pero también implica desafíos como el diseño, la implementación, el funcionamiento y el mantenimiento.

- **Directorio:**

  Cada servicio se enfoca en una función comercial específica y se puede implementar, escalar y actualizar de forma independiente. Este enfoque ofrece ventajas como la agilidad, la resiliencia, la escalabilidad y la facilidad de integración.

- **Bus de servicio:** El diseño de bus de servicio arquitecturas de microservicios es un patrón que permite la comunicación asíncrona entre servicios mediante el uso de un intermediario que transmite los mensajes entre los productores y los consumidores. Este patrón ofrece ventajas como el desacoplamiento, la fiabilidad, la escalabilidad y la tolerancia a fallos.

Conclusión.

Como conclusión tenemos que las arquitecturas de servicios, en este caso específicamente de los microservicios es de mucha utilidad y tiene diferentes propiedades que nos pueden ayudar a entender de mejor manera las especificaciones de esta y que además pueda ser de mucha utilidad para empresas pequeñas.

Bibliografía.

Autor desconocido, (03 de septiembre del 2019), Arquitectura de microservicios. Arquitectura de microservicios: qué es, ventajas y desventajas (decidesoluciones.es)

Autor desconocido, (19 de agosto del 2023), Estilo de arquitectura de microservicios. Estilo de arquitectura de microservicios - Azure Architecture Center | Microsoft Learn

Fracisco M. Garrido Jose L.(19 de agosto del 2023), Una arquitectura orientada a microservicios. (PDF) Una Arquitectura Orientada a Microservicios y Dirigida por Eventos para el Desarrollo de Sistemas de eSalud Avanzados: Caso de Evaluación de Fragilidad en Mayores (researchgate.net)

Autor desconocido, (03 de agosto del 2023), Equilibrio de carga en microservicios. Equilibrio de carga en microservicios con NGINX: guía detallada | AppMaster

Autor desconocido, (21 de noviembre del 2022), Arquitectura de microservicios. Arquitectura de microservicios | AppMaster

Autor desconocido, (19 de agosto del 2023), Arquitectura monolítica frente a microservicios. Arquitectura monolítica frente a microservicios: en qué se diferencian | AppMaster