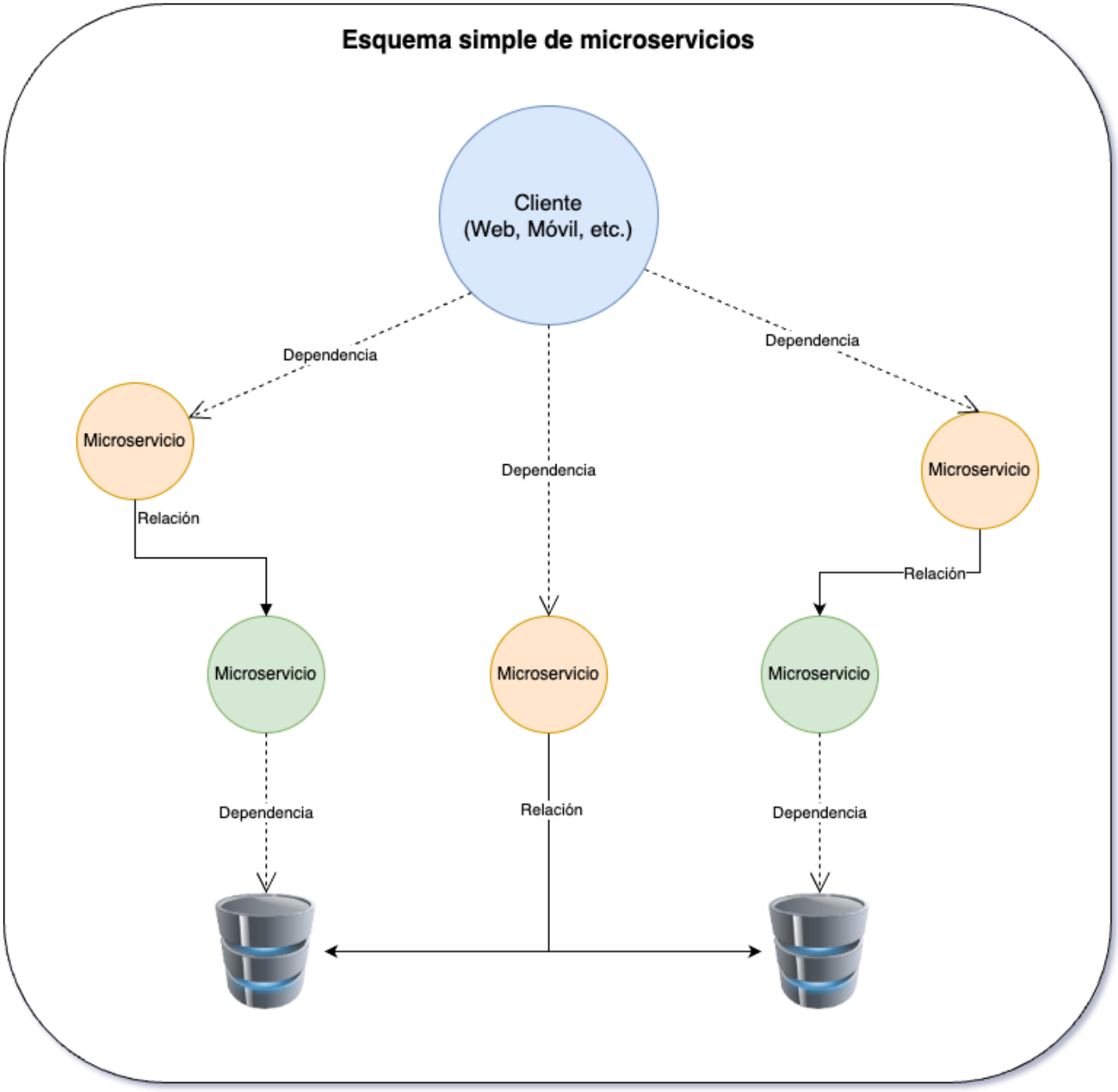
Los microservicios, también conocidos como arquitectura de microservicios, son un estilo de arquitectura que estructura una aplicación como un conjunto de servicios pequeños e independientes que se ejecutan en su propio proceso y se comunican entre sí mediante protocolos ligeros, generalmente HTTP o mensajería.

Cada servicio se enfoca en una funcionalidad específica del negocio y puede ser desarrollado, desplegado y escalado de manera independiente. Esto contrasta con la arquitectura monolítica tradicional, donde todas las funcionalidades están integradas en una única aplicación grande.



Analicemos por un momento el gráfico anterior, en el podemos observar como se detalla la relación y dependencia de los diferentes clientes de una aplicación y la relación y dependencia de sus microservicios de forma interna.

Lo importante que debemos entender es que cada microservicio contiene una parte de lógica del negocio o una funcionalidad muy importante en el sistema, por ejemplo uno de los

microservicios puede tratar únicamente de la gestión de usuario (registro, suscripciones e información de pago.). Mientras que otro microservicio puede ser el encargado de gestionar los resultados de búsquedas de un buscador de un sitio web, por ejemplo de un sitio de música tipo Spotify o YouTube Music (mi fav) donde la búsqueda es fundamental para garantizar una buena experiencia al usuario.

Características de los Microservicios

1. Descomposición por Funcionalidad

Tal y como expliqué antes la idea es que cada microservicio se enfoque en una funcionalidad específica del negocio, como gestión de usuarios, procesamiento de pagos o manejo de inventarios.

2. Desarrollo Independiente

Los microservicios pueden ser desarrollados por equipos diferentes utilizando distintos lenguajes y tecnologías, lo que facilita la adopción de la mejor herramienta para cada tarea. Por ejemplo podrías tener un servicio construido en Java con Spring Framework y otro construido en Python y Flask.

3. Despliegue Independiente

Cada microservicio puede ser desplegado de manera independiente, permitiendo actualizaciones y mejoras sin afectar a toda la aplicación. Piénsalo así si por alguna razón alguien hace un despliegue errado en un microservicio determinado, solo ese microservicio dejará de estar disponible pero las otras funcionalidades del sistema pueden seguir funcionando perfectamente.

4. Escalabilidad

Los microservicios permiten escalar solo las partes de la aplicación que necesitan más recursos, mejorando la eficiencia.

Ventajas de los Microservicios

1. Flexibilidad Tecnológica

Permiten a los equipos elegir la tecnología más adecuada para cada servicio, facilitando la

adopción de nuevas tecnologías y herramientas, pero cuidado con creer que con esto ya no vamos a vivir en un mundo lleno de software legacy, esto es y será así siempre por el costo de la mano de obra, así que aunque tengas la opción de crear cada microservicio con una tecnología diferente, ten cuidado en elegir una tecnología estable en el tiempo.

2. Escalabilidad Mejorada

Los microservicios pueden escalarse de forma independiente según las necesidades, lo que optimiza el uso de recursos, en teoría puedes desplegarlos como quieras, cada uno en una vm independiente, hay varias herramientas de orquestación de microservicios que te pueden ahorrar muchas tareas.

3. Despliegue Continuo

Facilitan el despliegue continuo y las actualizaciones frecuentes, mejorando el tiempo de comercialización. En este punto son muy superior a los monolitos, pero esa superior conlleva una serie de desafíos que exploraremos más adelante.

Desafíos de los Microservicios

Los microservicios tienen un desafío enorme y es por eso que no los recomiendo nunca como una primera opción a pesar de sus maravillosas ventajas, me refiero a la cantidad de recursos que hay que tener para gestionarlos de forma correcta, los costos de infraestructura, el personal, en fin. Cosas que no van para nada con equipos o empresas pequeñas.

Solo deberías considerar los microservicios si tienes suficiente dinero y recursos para hacer algo así, no hablo de pruebas de concepto esas las puedas hacer sin problema, me refiero a microservicios de un software empresarial, es más. Muchas veces ni es necesario porque son aplicaciones que van a usar un número ínfimo de personas y con un monolito lo puedes hacer aún mejor, más barato y hasta en menor tiempo.

Tengo todo un articulo hablando de los monolitos, puedes verlo haciendo click aquí.

Veamos un poco más a detalle los desafíos.

1. Complejidad Operacional

La gestión de múltiples servicios independientes puede ser compleja y requiere

herramientas adecuadas para el despliegue, monitoreo y escalabilidad.

2. Comunicación entre Servicios

La comunicación entre microservicios añade latencia y puede ser un punto de fallo si no se gestiona adecuadamente.

3. Consistencia de Datos

Mantener la consistencia de datos entre servicios puede ser complicado y puede requerir patrones avanzados como la consistencia eventual.

4. Gestión de Transacciones

Las transacciones distribuidas son más difíciles de gestionar en una arquitectura de microservicios.

Ejemplos de Uso

Siempre es importante observar el mercado para copiar cosas buenas y omitir cosas malas así que quise dejarte algunos ejemplos relevantes de microservicios.

1. Amazon

Amazon ha adoptado una arquitectura de microservicios para mejorar la escalabilidad y flexibilidad de su plataforma de comercio electrónico.

2. Netflix

Netflix utiliza microservicios para gestionar sus servicios de streaming, lo que les permite desplegar actualizaciones frecuentes y mejorar la disponibilidad.

3. Spotify

Spotify utiliza microservicios para gestionar su plataforma de música en streaming, facilitando la innovación y el despliegue continuo.

Tengo también un video en Youtube que explica un poco el tema, lo puedes ver aquí