Desafíos:

Escalabilidad

Concurrencia

Escalabilidad:

Cuando, por más que crece mi número de usuarios, el rendimiento de acceso a los datos no se ve alterado, no se vuelve más lento

Slashdotled:

Cuando una página web de un momento a otro se vuelve muy popular y el número de pedidos al servidor se incrementa exponencialmente, provocando que se sature.

Concurrencia:

Cuando se intenta acceder al mismo conjunto de datos en la base de datos por parte de varios de usuarios al mismo tiempo.

La sincronización se va en contra de la escalabilidad, porque consiste en darte un periodo de tiempo a cada usuario para acceder a cada conjunto de datos.

Relajar la concurrencia, en bases de datos no concurrentes, por ejemplo, si dos usuarios dan like a una foto, y pasa de tener 17 likes a 16 likes, no es un problema serio, porque likes en una foto no es algo importante de registrar en una base de datos.

Las transacciones con rollback & recovery, es ser optimista y permitir a los usuarios relizar transacciones, pero si son inconsistentes entre ellas, se realiza rollback y recuperamos la anterior información.

Paralelismo y concurrencia:

Paralelismo es cuando literalmente se ejecutan dos aplicaciones a la vez, (32 o 64 al mismo tiempo para computadoras actualmente), en cambio la concurrencia, varias aplicaciones se ejecutan turnándose en periodos de tiempo tan pequeños que nos parece que se ejecutan al mismo tiempo.

Instrucciones atómicas:

Son instrucciones que no pueden ser interrumpidas durante su ejecución.

Los microservicios son ineficientes, porque repiten estados.

Transparencia:

Componentes del sistema deben estar ocultos y parecer uno solo.

Transparencias de localización: Por ejemplo, no necesito saber dónde estará una persona al recibir un correo electrónico.

Entidades y paradigmas de comunicación:

Las unidades que se comunican se clasifican de acuerdo a:

Entidades de orientación de sistemas

Entidades de orientación de problemas

Cómo se comunican las unidades (niveles de abstracción):

Mediante comunicación de interprocesos, que es lo más simple.

Invocación remota, más avanzado en complejidad.

Comunicación indirecta, último y más avanzado nivel de abstracción

Un middleware es por ejemplo una cola de mensajes (MQ) que recibe los datos que envía un computador hacia un servidor, porque pasa a través del MQ, para que posteriormente este reenvié el mensaje a uno o varios servidores.

Modelo arquitectónico:

Se organizan en capas de software:

Aplicaciones y servicios

Middleware: Resuelve los problemas de transparencia, escalabilidad

Sistema operativo.

Hardware.

La combinación del sistema operativo con el hardware se conoce como plataforma.

Los middlewares son sistemas distribuidos.

Ejemplos de middleware para invocación remota de métodos:

RPC

SunRPC

JavaRMI

CORBA

Notificación de eventos es una abstracción que me permite a mi conocer cuando una actualización está disponible.

Ejemplos de notificación de eventos:

JNI

SNS

Servicios de Naming tenemos:

LDAP

Usar un middleware suele traer un tipo de overheal (sobrecarga) en el sistema.

Los problemas que me genera tener un solo servidor son:

Fácil saturación.

Punto único de fallo, si me falla el servidor, y es único, no puedo acceder a la base de datos, a diferencia de tener varios servidores bien estructurados, si falla uno o varios, puedo acceder a algunos datos a través de los servidores que están activos.