Capítulo 11: Escalabilidad

Víctor Iranzo

1 de junio de 2018

1. Introducción

Podemos hacer todo lo posible para evitar que un sistema falle, pero llego un punto en la escala, la probabilidad de fallo es inevitable. Muchas organizaciones invierten mucho esfuerzo en evitar que un fallo se produzca, pero muy poco en mecanismos para recuperar el sistema una vez se ha producido.

Los mecanismos de escalabilidad de nuestro sistema han de ser conformes a sus necesidades: tener un sistema capaz de reaccionar automáticamente al incremento de carga o a un fallo es fantástico pero exagerado para ciertos servicios. También es necesario cuantos fallos pueden tolerar o cuánto tiempo puede estar el sistema caído para nuestros usuarios. Estos requisitos se pueden definir así:

- Latencia o tiempo de respuesta: cuánto tiempo puede una operación tardar. La carga que sufre un sistema puede influir en su tiempo de respuesta.
- Disponibilidad: cuánto tiempo está disponible un servicio respecto del tiempo que se deseaba que estuviera en funcionamiento. ¿Puede estar el servicio caído o tiene que estar disponible 24/7?
- Durabilidad de los datos: cuánto tiempo se deben almacenar ciertos datos y cuánta pérdida de datos se considera aceptable.
- Resilencia: es la capacidad de un sistema para tolerar fallos y continuar trabajando. Un sistema que por culpa de un servicio caído deja de funcionar es menos resilente que un sistema que puede continuar ofreciendo el resto de sus funcionalidades.

Existen varios modos de fallo que se pueden dar cuando un sistema está caído. Responder de forma lenta es uno de los peores porque aumenta la latencia de todas las peticiones para responder que el servicio está caído. Esto puede producir fallos en cascada dentro de la cadena de invocaciones.

2. Gestión de fallos en el sistema

En el libro "Antifragile"se explica como una organización como Netflix basada en la infraestructura de AWS prueba la tolerancia a fallos de sus servicios incitando al fallo de estos. Para ello, emplea un conjunto de programas que componen el "ejército de simios". Chaos Monkey se encarga durante ciertas horas al día de apagar máquinas de forma aleatoria. Chaos Gorilla hace una función similar pero con los centros de datos. Latency Monkey simula redes de bajo rendimiento.

Todos estos fallos pueden darse en producción y esta librería es una buena manera para comprobar si se está preparado para tolerarlos.

2.1. Timeouts

2.2. Cortocircuitos

2.3. Bulkhead

2.4. Aislamiento

3. Idempotencia

4. Técnicas de escalabilidad

4.1. Escalabilidad vertical

4.2. Dividir la carga de trabajo de un servicio

4.3. Balanceadores de carga

4.4. Sistema basado en trabajadores

5. Escalabilidad en bases de datos

5.1. Escalabilidad en la lectura

5.2. Escalabilidad en la escritura

5.3. CQRS

6. Caching

6.1. Tipos de caching

7. Autoescalado

- Escalado reactivo:
- Escalado predictivo:

8. El teorema de CAP

8.1. Sistemas AP

8.2. Sistemas CP

8.3. Balance entre sistemas AP y sistemas CP

9. Descubrimiento de servicios

9.1. DNS

9.2. Herramientas para el descubrimiento de servicios

- Zookeeper
- Consul
- Eureka

10. Documentación de servicios

- Swagger
- HAL

11. Descripción de un servicio