

DASE: Actividades Tema 3

Actividad 1

En las páginas 9 a 11 de la presentación de esta unidad se explica la definición de escalabilidad propuesta por Jogalekar y Woodside. Utilice las expresiones presentadas en esas páginas e indique si cada uno de los sistemas descritos en los próximos apartados podría considerarse un sistema escalable o no. Razone por qué. En caso de que el sistema no pueda considerarse escalable, indique si existe algún rango de cargas en que su escalabilidad fuera razonable.

1. Sistema de almacenamiento de datos para un servicio de comercio electrónico, empleando replicación de datos basada en certificación con consistencia secuencial. Se definen los siguientes parámetros y variables:
 - k_1 : Número de usuarios (Para analizar la escalabilidad, asuma que varía de 10000 en 10000 usuarios).
 - $x_1(k_1)$: Número de registros en la base de datos.
 - $x_1(k_1) = 2000 + 1.2 * k_1$
 - $x_2(k_1)$: Número de nodos en los que se replica la base de datos.
 - $x_2(k_1) = 1 + \lfloor k_1 / 10000 \rfloor$
 - Cada servidor (nodo) puede atender casi 22 peticiones por segundo en el mejor escenario posible. En función de k_1 su rendimiento (expresado en peticiones por segundo) es:
 - $T(k_1) = 22 - x_2(k_1) - x_1(k_1) / 20000$Se asume que los k_1 usuarios generan precisamente esa carga.
 - El beneficio promedio en cada operación depende del número de nodos (variable " $x_2(k_1)$ "), pues cuantas más réplicas haya éstas se distribuyen geográficamente y siempre se utiliza la réplica más cercana a cada usuario, reduciendo el tiempo de respuesta y aumentando la satisfacción del usuario, que se animará así a realizar nuevas peticiones. La fórmula para calcular cuánto (en euros) se ingresa por cada petición realizada es:
 - $0.02 + 0.001 * x_2(k_1)$
 - El alquiler de cada servidor cuesta 2 euros por día.

2. Sistema similar al anterior, pero utilizando consistencia final y permitiendo que cualquier réplica sirva cualquier operación. Esto reduce las necesidades de bloqueo y permite propagación perezosa de las modificaciones. Se definen los siguientes parámetros y variables:
- $k1$: Número de usuarios (Para analizar la escalabilidad, asuma que varía de 10000 en 10000 usuarios).
 - $x1(k1)$: Número de registros en la base de datos.
 - $x1(k1) = 2000 + 1.2 * k1$
 - $x2(k1)$: Número de nodos en los que se replica la base de datos.
 - $x2(k1) = 1 + \lfloor k1/10000 \rfloor$
 - Cada servidor (nodo) puede atender casi 24 peticiones por segundo en el mejor escenario posible. En función de $k1$ su rendimiento (expresado en peticiones por segundo) es:
 - $T(k1) = 24 - x2(k1)/2 - x1(k1)/2000000$Se asume que los $k1$ usuarios generan precisamente esa carga.
 - El beneficio promedio en cada operación depende del número de nodos (variable " $x2(k1)$ "), pues cuantas más réplicas haya éstas se distribuyen geográficamente y siempre se utiliza la réplica más cercana a cada usuario, reduciendo el tiempo de respuesta y aumentando la satisfacción del usuario, que se animará así a realizar nuevas peticiones. La fórmula para calcular cuánto (en euros) se ingresa por cada petición realizada (cambia respecto al apartado anterior) es:
 - $0.02 + 0.0002 * x2(k1)$
 - El alquiler de cada servidor cuesta 2 euros por día.

Actividad 2

Las expresiones utilizadas en la definición de escalabilidad dada por Jogalekar y Woodside consideran tanto el rendimiento obtenido como el coste económico.

a) Justifique por qué debe considerarse relevante el aspecto económico al evaluar la escalabilidad de un sistema. Tenga en cuenta que no todos los sistemas distribuidos escalables han sido sistemas de computación en la nube.

b) Si se descartaran los costes, indique qué expresiones deberíamos utilizar para evaluar la escalabilidad de un sistema distribuido. Aplique las fórmulas resultantes a los ejemplos de la actividad 1 e indique si hay variaciones significativas en los rangos de carga para los que cada sistema puede considerarse escalable.

Actividad 3

A la hora de incrementar la escalabilidad de un sistema se han presentado en esta unidad ciertos mecanismos básicos: (a) reparto de tareas, (b) reparto de datos, (c) replicación, (d) uso de cachés. El objetivo general de todos ellos es asegurar que cada proceso servidor sea responsable de un subconjunto de la carga global y que las necesidades de comunicación entre ellos disminuyan, reduciendo así la necesidad de sincronización y bloqueo.

No se ha discutido nada sobre la comunicación. Asumamos que en un sistema existen los siguientes mecanismos de comunicación:

1. Canales punto a punto quasificables (que garantizan la entrega de los mensajes siempre que no falle ni el emisor ni el receptor) sobre los que se implanta comunicación no persistente sincrónica.
2. Canales punto a punto quasificables implantados por un middleware capaz de proporcionar comunicación persistente sincrónica.
3. Conector proporcionado por un middleware que difunde un mensaje a un grupo de procesos (a las réplicas del servicio solicitado, por ejemplo), respetando orden total en la entrega de los mensajes (todos los destinos reciben todos los mensajes en un mismo orden). Se proporciona comunicación no persistente sincrónica. El middleware necesitará utilizar un protocolo en el que todos los procesos intercambien múltiples mensajes, empleando un tiempo no despreciable (entre 2 y 3 ms en una red local) para realizar la difusión ordenada.
4. Como en el caso anterior (caso 3), pero ahora la comunicación proporcionada es no persistente asincrónica.

Discutir qué ventajas o inconvenientes aportaría cada uno de los mecanismos de comunicación descritos a la hora de mejorar la escalabilidad del sistema. Es decir, indicar cómo influye cada mecanismo de comunicación a la hora de reducir la sincronización y bloqueo entre los procesos del sistema. Según este análisis, ¿qué factores, de entre los siguientes, aportan alguna mejora? ¿Por qué?

- a) Que la comunicación sea asincrónica.
- b) Que la comunicación sea persistente.
- c) Que sea posible dirigir los mensajes a un grupo de destinatarios, garantizando su entrega en los procesos que no hayan fallado.

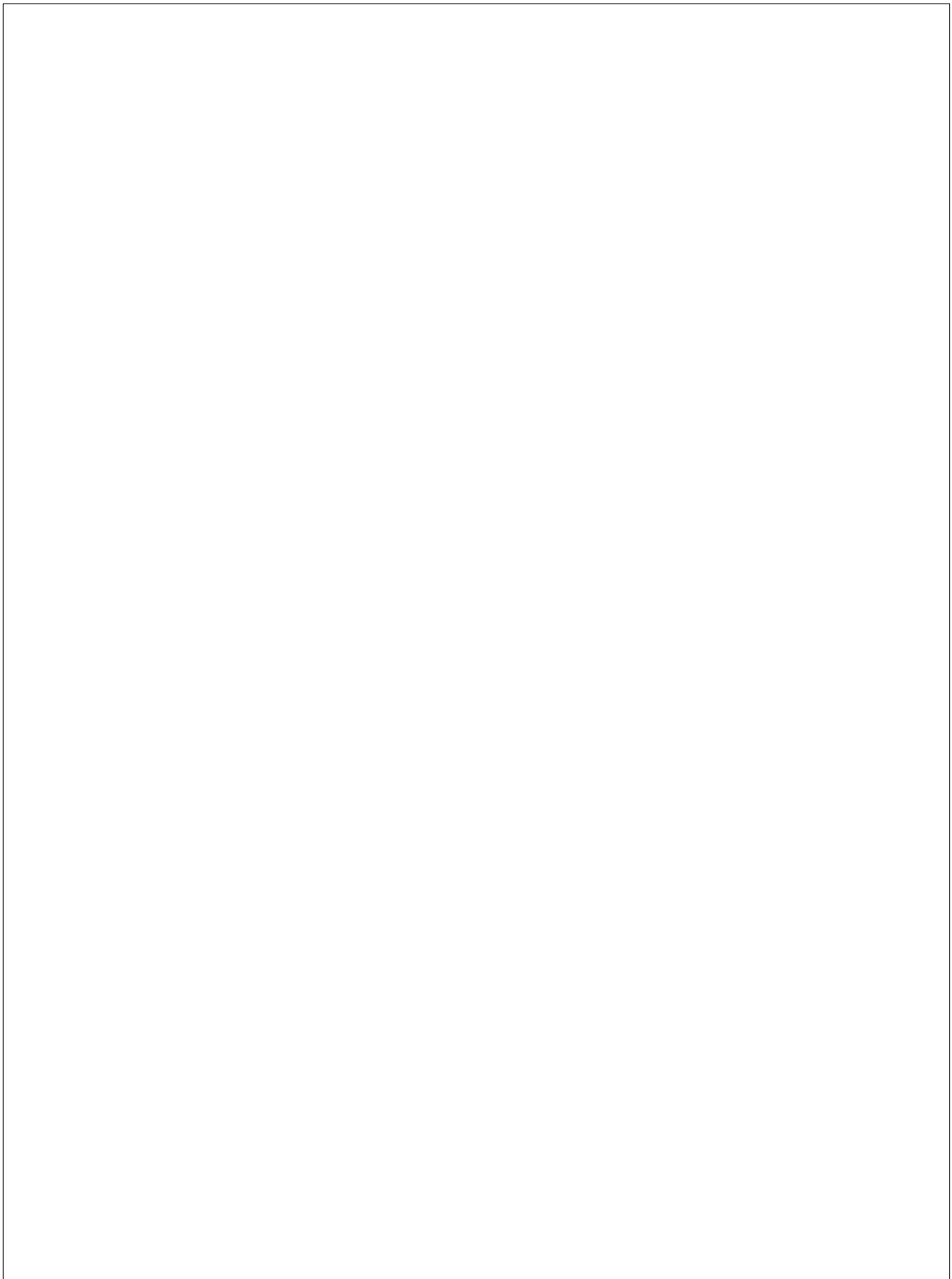
NOTA: Se asumen los siguientes conceptos:

Comunicación sincrónica: la que bloquee al emisor hasta que el sistema de comunicaciones asegure que el mensaje podrá entregarse a sus receptores.

Comunicación asincrónica: la que no bloquea jamás al emisor.

Comunicación persistente: aquella en la que el sistema de comunicaciones será capaz de mantener el mensaje aunque alguno de los extremos de comunicación no esté activo. El servicio de correo electrónico emplea este tipo de comunicación: cuando se envía un correo el receptor no tiene por qué tener ningún proceso activo capaz de recibir el mensaje.

Comunicación transitoria (o no persistente): aquella en la que un mensaje no puede ser transmitido mientras no estén los dos extremos de comunicación activos.



Actividad 4

Existen varios algoritmos clásicos de exclusión mutua para sistemas distribuidos (centralizado, distribuido, para anillos... cuyas descripciones básicas pueden consultarse en el fichero `exclusiónMutua.pdf` que acompaña a este boletín). Revise con cuidado esos algoritmos e indique al menos una razón por la que desaconsejar el uso de cada uno de ellos en un sistema escalable, además de las que ya se mencionan en esas descripciones. Recuerde que una solución ideal debería minimizar el intervalo de suspensión de cualquier proceso que intentase acceder a una sección crítica.

Actividad 5

Algunos sistemas distribuidos recurren a la centralización para realizar ciertos servicios breves de manera eficiente, reduciendo el tráfico de mensajes necesario (si se compara esa solución con otra descentralizada). Estudie los siguientes ejemplos e indique si esa operación “centralizada” pone en peligro la escalabilidad del sistema que la utiliza. Para ello analice si se podría manejar un número potencialmente alto de peticiones por cada unidad de tiempo. Considere también el tiempo necesario para atender una petición en esa aplicación o servicio.

1. En las primeras generaciones de sistemas P2P la localización de recursos era realizada por un servidor central (posiblemente replicado para superar las situaciones de fallo). En ese tipo de sistemas, los usuarios de la aplicación P2P facilitan el nombre del fichero a localizar y el servidor retorna las direcciones de varios nodos que mantienen copias de ese fichero. Posteriormente, la aplicación se encarga de contactar con esos nodos y descargar el fichero desde ellos. La tabla que guarda la correspondencia entre nombres de fichero y direcciones de los nodos que los guardan cabe perfectamente en la memoria principal del servidor.

2. En las primeras generaciones de sistemas gestores de bases de datos distribuidas, cuando una transacción solicitaba su finalización un nodo coordinador utilizaba un protocolo de *commit* distribuido de dos o tres fases. En cada fase, el coordinador espera los “votos” de los nodos que han gestionado alguna de las subtransacciones y guarda en disco el estado de la transacción. En función de los votos recibidos, el coordinador decide si la transacción puede terminar con éxito o no y lo comunica a los participantes para iniciar una nueva fase o finalizar el protocolo. Las fases introducen etapas de sincronía: ningún nodo puede empezar una nueva fase mientras alguno de los demás no haya terminado la actual.

3. En el modelo de replicación pasivo, una de las réplicas tiene un rol primario y recibe todas las peticiones que impliquen una modificación del estado del servicio. El resto de réplicas tiene un rol secundario y, en algunos casos, se admite que procesen directamente aquellas peticiones de solo lectura. La réplica primaria puede realizar todo el control de concurrencia mediante mecanismos locales. Al finalizar cada petición que haya modificado el estado, propagará de manera asincrónica las modificaciones a las réplicas secundarias, que las aplicarán en orden. Un mismo conjunto de máquinas puede ejecutar un alto número de servicios bajo este modelo de replicación, asignando cada servicio el rol primario a una réplica diferente.