# Pontificia universidad católica de chile

# Escuela de ingeniería

# Departamento de ciencia de la computación

# 26 de Septiembre de 2019

# IIC2173 Arquitectura de Sistemas de Software

##### Interrogación 1

##### Instrucciones: Sea preciso, no es necesario escribir mucho pero si ser preciso. En caso de ambigüedad use su criterio y agregue el supuesto en la respuesta correspondiente.

1. (2.0pts) Respecto a los siguientes patrones arquitectónicos indique cuáles son los atributos de calidad que se ven afectados:

a) (0.5) Máquina virtual

RNF +:

RNF -:

b) (0.5) Pipe & Filter

* RNF +:
* RNF -:

c) (0.5) Publisher/Subscriber

RNF +:

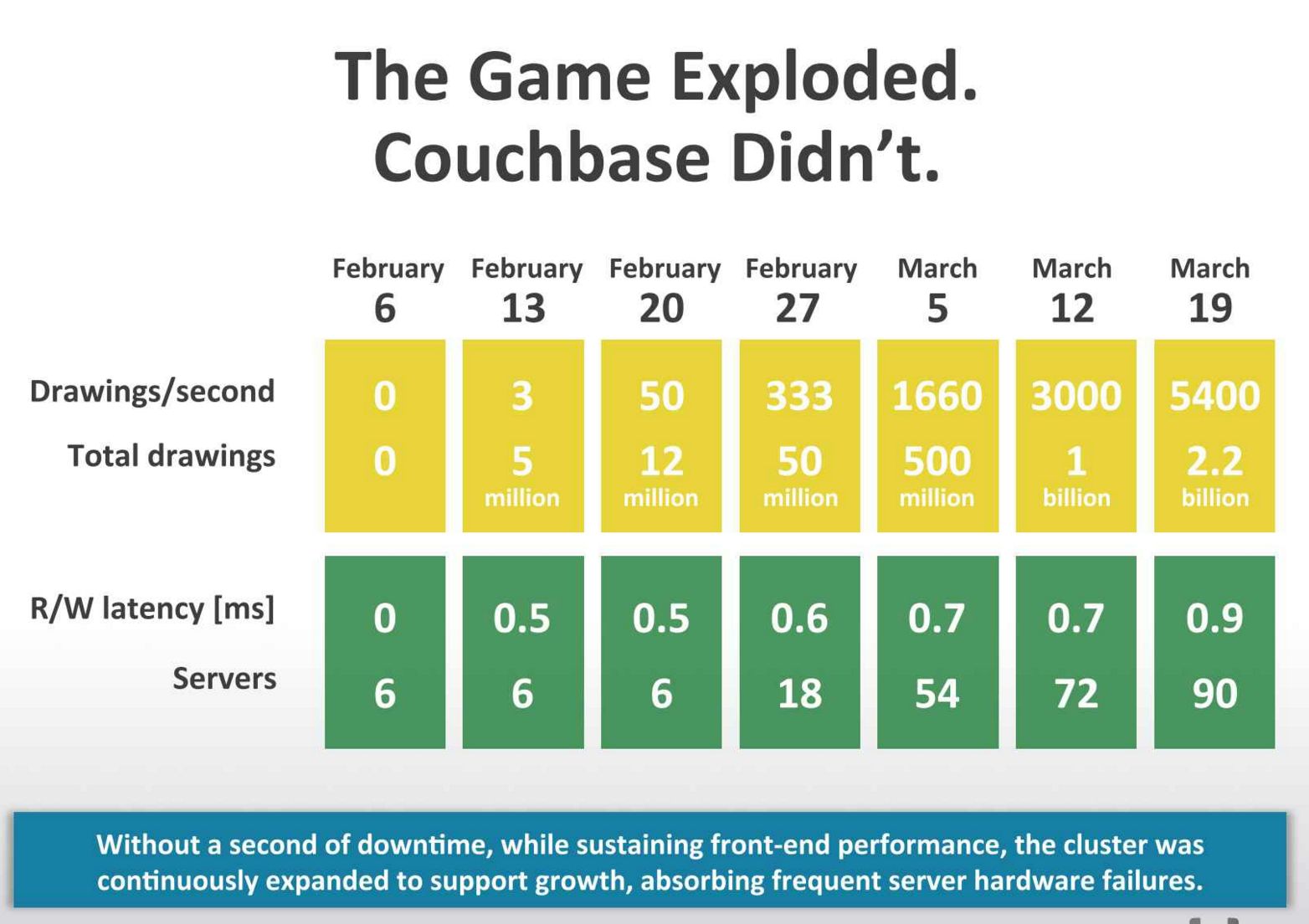
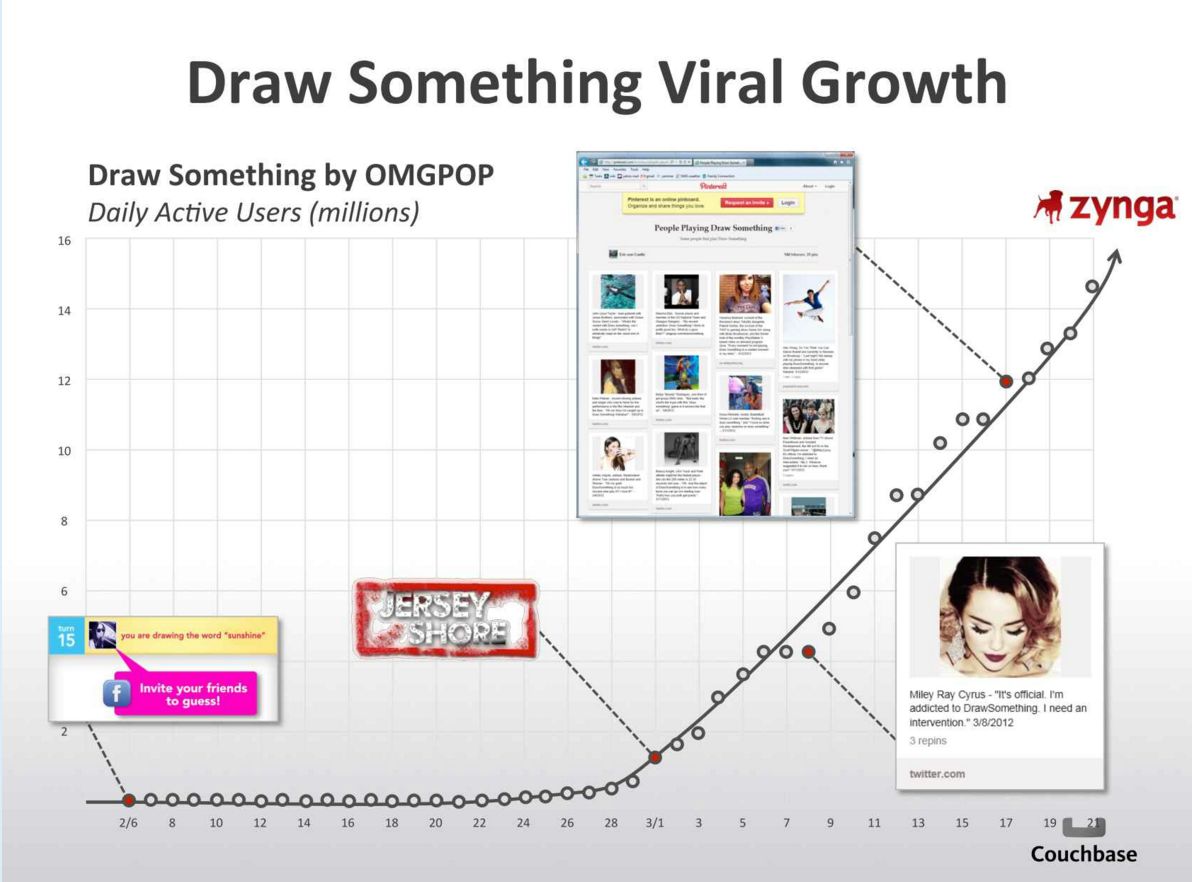
RNF -:

* d) (0.5) Código móvil
* RNF +:
* RNF -:

1. (1.0pts)Caso *DrawSomething*

* *DrawSomething* fue un juego social para dispositivos móviles desarrollado por OMGPop. El juego, basado en P ictionary, consiste en:

1. uno de los jugadores recibe una palabra
2. ese jugador hace un dibujo en base a dicha palabra
3. al terminar, el dibujo se envía al otro jugador
4. el otro jugador debe adivinar la palabra inicial en un tiempo dado

* En la imagen adjunta se ve la evolución de usuarios activos en la aplicación. Desde su lanzamiento, en 50 semanas llegaron a más de 50 millones de usuarios activos (sin *downtime*). En la tabla de información se muestra que para el 19 de marzo recibían 5.400 dibujos por segundo manejados por un total de 90 servidores con una latencia de lectura/escritura de 0.9ms. (Zynga lo compró en 180 millones de dólares debido a su éxito)
* Inicialmente la arquitectura consistía en clientes nativos (para los distintos dispositivos móviles), y una capa de servidores para manejar los *requests* alimentados por un *load­balancer* (*HAProxy*) . La aplicación en el servidor era muy pequeña para lograr tener tiempos de respuesta tan bajos como fuese posible. El almacenamiento era basado en una base de datos montada sobre *Amazon Web Services.*
* Posteriormente se agregaron varias mejoras en la aplicación para manejar de mejor manera los *requests* simultáneos. Por ejemplo, reemplazaron el sistema de bases de datos por *CouchBase* (NoSQL, documentos) ­ que permite escalabilidad horizontal ­ sobre discos de SSD. Otras medidas para escalar el sistema fueron implementadas en las siguientes semanas.
* 
  1. (0.1pts) ¿Cuál es el principal estilo arquitectónico implementada? Cliente Servidor
  2. (0.3pts) Una forma económica y eficiente de enfrentar altos niveles de carga es mediante la utilización de caché. ¿Por qué cree Ud. que no se menciona como un factor relevante?

Esto se debe principalmente a que los clientes ya tienen toda la información que necesitan regularmente y sólo deben acceder a los datos una vez. Los “POST” no se cachean.

* 1. (0.3pts) Cuando se realizó el cambio de la base de datos, también se llevó a cabo una “migración perezosa” de datos (*lazy migration*). Ésto es que se copia una referencia a los datos originales, sin embargo solo son copiados a la nueva fuente de datos cuando se consulta por ellos. ¿Qué beneficio se obtiene de este patrón?

Este patrón ayuda a mantener la consistencia de datos sin la necesidad inmediata de realizar la transferencia masiva de datos al nuevo sistema, permitiendo así que el sistema no tenga “*downtime*” y dedicarse principalmente a las necesidades de los usuarios.

* d. (0.3pts) A continuación se muestran triples extraídos de la tabla de información. Estos están en el formato (<dibujos por segundo>, <latencia en ms>, <#servers>). ¿Qué puede decir acerca de la siguiente secuencia: [(3, 0.5, 6), (50, 0.5, 6), (333, 0.6, 18), (3000, 0.7, 54), (5400, 0.9, 90)]?

Al principio no hubo cambios en la infraestructura y estaban preparados para esa carga. Factores relevantes a mencionar es que la cantidad de servidores utilizados por el sistema crece mucho más lento que el volumen de *requests* manejados por el sistema. Y por otra parte es importante mencionar que la latencia con la que se manejan los *requests* se mantiene muy baja a pesar del explosivo incremento en la carga del sistema.

1. (1.0pts) Considere un emprendimiento que se ha especializado en un rubro en particular donde requiere almacenar información no estructurada. El negocio es bastante puntual, por lo que la escala es pequeña y según su planificación de crecimiento se espera tener hasta tres servidores para la aplicación y uno para la base de datos para los próximos cuatro años.

Acuden a un (supuesto) experto quien les recomienda las siguientes opciones para utilizar como motor de bases de datos. La primera es PostgreSQL ya que es un motor relacional clásico muy utilizado con una gran comunidad, las otras recomendaciones son motores no relacionales con fuertes propiedades distribuidas. MongoDB es fuertemente consistente y por CouchDB promueve la disponibilidad de los datos.

Para confirmar lo sugerido, le preguntan a Ud. por su opinión. En particular se espera justifique brevemente con al menos 1 beneficio y contra-indicaciones (si aplican) cada una de las tres recomendaciones entregadas por el experto anterior.

Por último se espera entregue su propia recomendación (entre las opciones entregadas).

PostgreSQL: motor relacional rápido, modelo conocido más ampliamente. Sin embargo el modelo a implementar para información no estructurada se vuelve demasiado complejo y tiende a bajar su performance.

CouchDB y MongoDB son motores no relaciones por lo cual se adecúan bastante a la tarea en cuestión. Ambos son buenas recomendaciones.

Es relevante que mencionen que las características de consistencia y disponibilidad solo salen a luz cuando el sistema de bases de datos se encuentra distribuido, lo cual dice en el enunciado que no es el caso.

\* al haber solo un nodo, se guarda en disco y al consultar es inmediatamente consistente y no pierde tiempo sincronizando con otros nodos por lo cual está disponible al instante

1. (2pts) Considere el sistema de “entretenimiento abordo” de un avión de última generación, el cual Ud. está ayudando a desarrollar. Cada pasajero contará con un monitor y un control alámbrico para poder interactuar con los diferentes servicios del sistema.

Previo al despegue, todos los dispositivos muestran un video tutorial que expone las instrucciones para actuar en algún caso de emergencia (el cual no puede ser interrumpido). Posterior a ésto, una vez que el avión haya despegado, el pasajero debe ser capaz de navegar por la lista de música o de videos (películas y series). Cada una de estas listas está categorizada tanto por su “género” como por su antigüedad en el sistema. Las listas son actualizadas periódicamente, función que es ejecutada durante la estadía en tierra (entre vuelos). Además, el pasajero puede interactuar con la reproducción de música o video en el momento que desee, así como adelantarlo, pausarlo y evaluarlo. Adicionalmente el pasajero puede, en cualquier instante del trayecto, acceder a información general del avión y del vuelo, la cual está siendo actualizada constantemente por el mismo avión.

Por otra parte, el capitán (piloto del vuelo) puede hablarles a todos los pasajeros en el momento que él estime conveniente, interrumpiendo la reproducción de música, videos y entrega de información, con el fin de que todos puedan prestar atención al mensaje.

1. (0.6pts) Indique (al menos 2) atributos de calidad que Ud. considere son fundamentales en este sistema. Enumérelos en orden de prioridad y justifique.
2. (0.6pts) Indique los patrones arquitectónicos (al menos 2) que va a emplear para solucionar el problema. Indique qué aspecto del problema ataca cada patrón (responsabilidad).
3. (1.0pts) Haga un diagrama de bloques indicando los componentes arquitectónicos y sus relaciones que reflejen una solución al problema planteado. Añada tanto detalle como haga falta para que quede claro cómo el patrón soluciona el problema planteado.
4. (0.3pts) Para cada conector, indique el tipo, dimensión, sub-dimensión y su valor.

(Hint: apóyese en el apéndice entregado)

1. (0.3pts) ¿Qué atributos de calidad (al menos 1) se ven perjudicados en su solución? Justifique
2. (0.2pts – Bonus) Si es que además se quiere considerar que existen juegos en la plataforma donde el pasajero puede jugar solo o desafiar y jugar en línea con otro pasajero. ¿Cómo se modificaría el diagrama anterior?