Práctica 5, 2ª parte - Servicio de almacenamiento basado en primario/copia

Daniel Rueda Macías

15 de enero de 2017

Introducción

En esta práctica se va a implementar un servicio de almacenamiento distribuido clave/valor utilizando el servicio de vistas implementado de la parte anterior volviendo a utilizar la plataforma de desarrollo Elixir/Erlang.

Comportamiento

En esta sección se va a describir el comportamiento de los nodos que intervienen en la vista del sistema de almacenamiento, es decir, los nodos primario y copia. El comportamiento general viene descrito por el diagrama de secuencia dado en el anexo. No obstante, si se ve mal se adjunta en la carpeta junto a los archivos de código.

Ambos nodos coinciden en que mandan latidos al gestor de vistas y éste les responde con la vista tentativa. El primario comprueba si la operación que le llega es la misma que la última realizada, esto evita la repetición de operaciones ante caídas de red.

Comportamiento del nodo primario

Hay que puntuar que se encarga de comprobar si hay un nodo copia nuevo, en caso afirmativo, le envía la BBDD para que la cargue y así ambos sean consistentes.

El nodo primario recibe una orden de realizar una operación por el cliente. Dicha operación puede ser:

- Leer una clave de la BBDD: El primario manda la operación de lectura al copia y espera una respuesta un tiempo determinado. Si la respuesta es que todo ha ido bien, efectúa la lectura, se guarda esta operación junto con el resultado y se lo envía al cliente. En caso contrario o si ha pasado más tiempo del establecido se informa al cliente de que ha habido un error.
- Escribir en una clave de la BBDD, el comportamiento es idéntico ya sea hash o escritura genérica, solo cambia la llamada a la función de escritura: El primario manda la operación al copia y espera una respuesta un tiempo determinado. Si la respuesta es que todo ha ido bien, efectúa la escritura ya sea hash o normal según corresponda, se guarda la operación y envía al cliente una confirmación. En caso contrario o si ha pasado más tiempo del establecido informa al cliente de que ha habido un error.

Comportamiento del nodo copia

Obedece a una orden más que el primario, la cual es cargar en su BBDD la que le envía el primario si el nodo acaba de ser nombrado como copia.

Respecto de las funcionalidades comunes al primario son casi iguales, salvo que las órdenes son recibidas del primario, no del cliente y no tiene que replicar la operación a nadie:

- Leer una clave de la BBDD: El nodo copia recibe la operación de lectura del primario, lee de la BBDD y se guarda la última operación junto con el resultado. Por último, manda una confirmación al primario.
- Escribir en una clave de la BBDD, el comportamiento es idéntico ya sea hash o escritura genérica, solo cambia la llamada a la función de escritura: El nodo

copia recibe una petición de escritura del primario, este escribe en su BBDD según el tipo de escritura que corresponda, guarda la última operación y envía una confirmación al primario.

Validación Experimental

Se han ejecutado las pruebas depurando los fallos sacando trazas por pantalla hasta que se pasaron todos los test. Algunos de los errores cometidos fueron:

- No cambiar el envío de la respuesta al cliente gestor de vistas por servidor de almacenamiento.
- Un error que no se había encontrado en la parte anterior sobre el gestor de vistas. Y es que siempre que se llegaba al número establecido de latidos fallidos, se promocionaba el nodo copia a primario. La solución no era siempre esta, por tanto, para remediarla se contabilizan los latidos de primario y del copia por separado. Si alguno de los dos o ambos manda un latido se pone su respectiva cuenta a 0. Así se pueden distinguir os siguientes casos de fallo.
    - Que el nodo copia falle y el primario no. En este caso se pasa a una nueva vista sólo con el primario.
    - Que el nodo primario falle y el copia no. Para corregir este fallo, se pasa a una nueva vista con el copia promocionado a primario y sin ningún nodo copia.
    - Que los dos fallen. En este caso se reseteaba la vista.

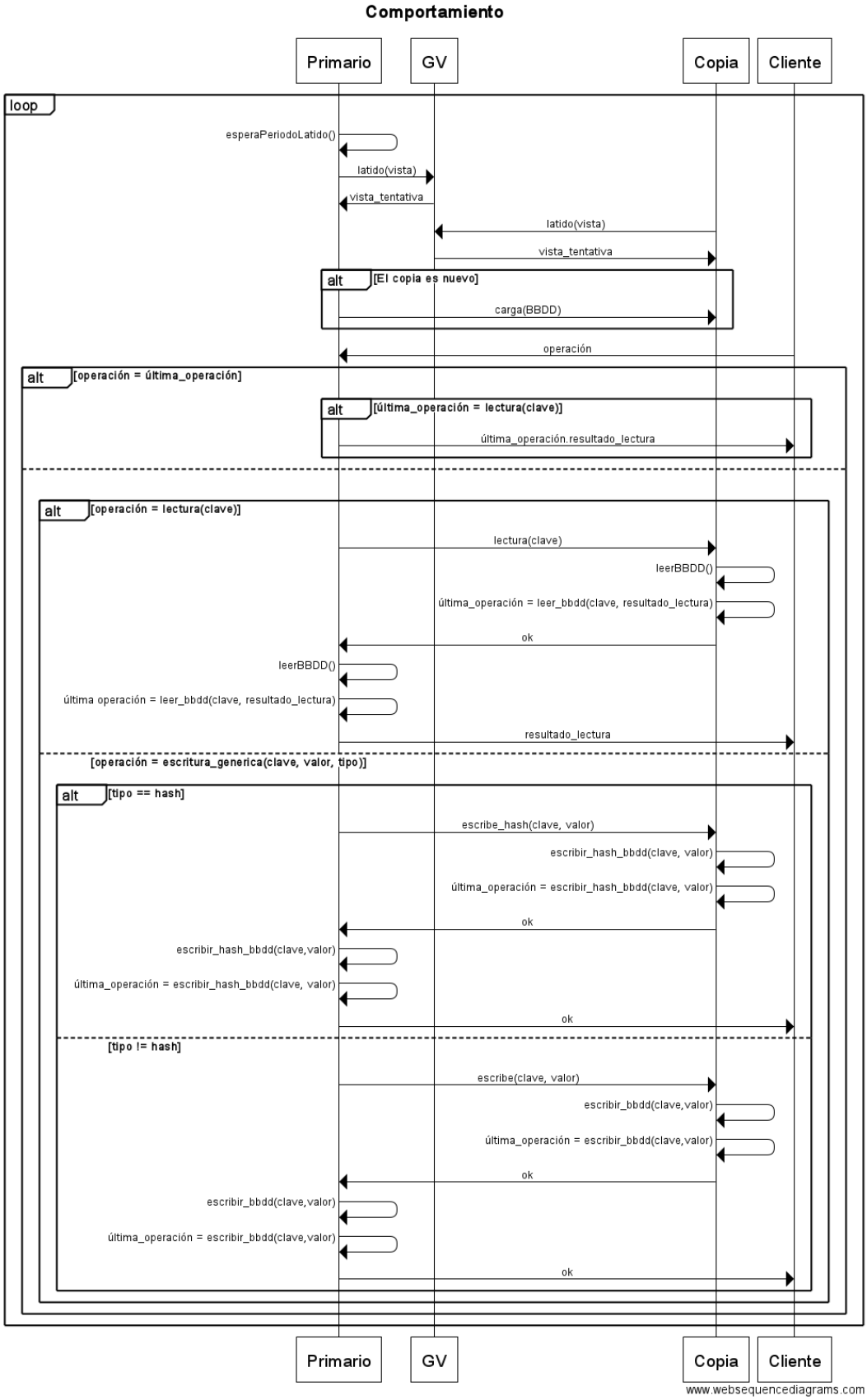
Una vez corregidos todos estos errores, los test fueron pasados a la perfección.

Conclusiones

En esta práctica se ha implementado un servicio de almacenamiento distribuido calve/valor tolerante a fallos utilizando el servicio de vistas de la parte anterior mediante la plataforma de desarrollo Elixir/Erlang.

No obstante, se sigue teniendo la limitación de que se tiene un solo gestor de vistas y que si éste cae también lo hace todo el sistema. Por tanto, para una mayor tolerancia a fallos se podría aplicar la replicación al igual que en el servicio de almacenamiento junto con los algoritmos de elección de líder aprendidos en clase.

Anexo


Comportamiento
Primario
GV
Copia
Cliente
loop
esperaPeriodoLatido()
latido(vista)
vista_tentativa
latido(vista)
vista_tentativa
alt
[El copia es nuevo]
carga(BBDD)
operación
alt
[operación = última_operación]
alt
[última_operación = lectura(clave)]
última_operación.resultado_lectura
alt
[operación = lectura(clave)]
lectura(clave)
leerBBDD()
última_operación = leer_bbdd(clave, resultado_lectura)
ok
leerBBDD()
última operación = leer_bbdd(clave, resultado_lectura)
resultado_lectura
[operación = escritura_generica(clave, valor, tipo)]
alt
[tipo == hash]
escribe_hash(clave, valor)
escribir_hash_bbdd(clave, valor)
última_operación = escribir_hash_bbdd(clave, valor)
ok
escribir_hash_bbdd(clave,valor)
última_operación = escribir_hash_bbdd(clave, valor)
ok
[tipo != hash]
escribe(clave, valor)
escribir_bbdd(clave,valor)
última_operación = escribir_bbdd(clave,valor)
ok
escribir_bbdd(clave,valor)
última_operación = escribir_bbdd(clave,valor)
ok
Primario
GV
Copia
Cliente
www.websequencediagrams.com