SSDD

Práctica 4

Diciembre 2018

Pablo Orduna Lagarma y Daniel Naval Alcalá

Introducción

En esta práctica hay que construir un servicio distribuido de almacenamiento que cuente con tolerancia a fallos con estado, utilizando el sistema de replicación primario/copia.

Para ello hay que diseñar e implementar el servicio gestión de vistas que supervisa los nodos y los analiza, así si uno de ellos no está respondiendo, automáticamente se reemplaza su lugar para que el cliente pueda seguir operando. (Trabajo a realizar en la práctica 5)

Los diferentes nodos pueden ser asignados a tres diferentes puestos:

- **Primario:** Es la primera réplica, las peticiones de escritura de clientes siempre se realizan de forma efectiva en el primario, y este, posteriormente escribe en las copias de seguridad.
- **Copia:** Son un refuerzo del primario y si el primario cae, una de las réplicas, que funcionaba como copia de seguridad, se convierte en primario.
- **En espera:** Son nodos que se están inactivos mientras el sistema gestor de vistas no detecta una caída o promoción de un nodo copia.

El funcionamiento normal se basa en una comprobación continua de que los nodos se encuentran operativos, esto se hace mediante un latido periódico que los nodos emiten cada 50 ms. En caso de que el gestor de vistas no reciba uno de esos latidos, el nodo que no responda será reemplazado de forma inmediata y se sustituirá en la nueva vista tentativa hasta que pueda ser validada.

Si el primario cae, la copia ocupará su lugar y un nodo en espera ocupará el lugar de la copia; en caso de que no haya una copia disponible, el gestor de vistas no podrá continuar debido a que no tiene un respaldo que asegure la integridad de los datos.

Diseño

El servidor está programado para recibir 3 tipos de mensajes: latidos (:latido), peticiones de obtención de la vista válida (:obten_vista) y mensajes internos para el manejo de las caídas por latidos fallidos (:procesa_situacion_servidores).

:latido

El manejo de los latidos en el servidor es la parte más compleja del desarrollo, principalmente debido a la enorme cantidad de casos posibles que pueden tener lugar en el escenario planteado.

En un primer lugar se ha dividido este apartado atendiendo al valor del número del latido, clasificando dichas partes en:

- Latido con número 0: significa que el nodo ha arrancado, ya sea por primera vez o tras una caída.

Primero se comprueba que existe nodo primario en la vista tentativa, de manera que, si no existe nos podemos encontrar con el caso de que sea la vista inicial o de que se hayan caído tanto el nodo primario como el nodo copia simultáneamente. También se tiene en cuenta el caso de que no exista un primario, pero si una copia, el cual tiene lugar si se introdujese un nuevo nodo tras la ejecución de la novena prueba.

Por el contrario, si el nodo primario existe se comprueba la existencia del nodo copia y se compara el nodo emisor con el nodo copia en el caso de que esté definido, diferenciando así entre todos los casos posibles.

- Latido con número diferente de 0: cuando el número del latido es diferente de 0 se han hecho también numerosas comprobaciones: identificación del nodo emisor como copia o primario de la vista tentativa, comparación del número de vista con el número enviado en el latido y la existencia del nodo actual en la lista de espera.

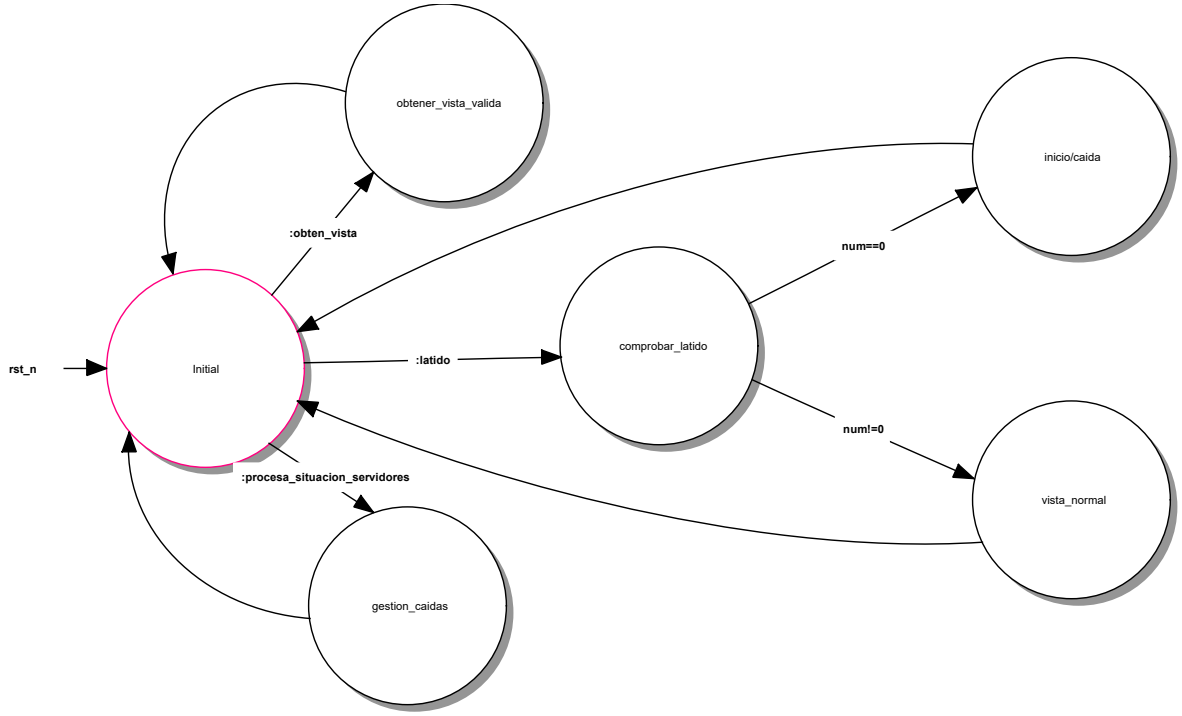
De esta manera se tienen en cuenta diferentes situaciones entre las cuales destacan, la recuperación de un nodo que había quedado aislado en una partición de red y la promoción de un nodo a copia, en caso de que la copia no se encontrase definida en la vista tentativa.

:obten_vista

El servidor del Gestor de Vistas recibe este mensaje por parte de un nodo cuando dicho nodo quiere obtener el valor de la vista válida, junto con el cual le avisa de si coincide con el de la vista tentativa.

:procesa_situacion_servidores:

Es un mensaje interno del propio proceso del gestor que se recibe cada 50ms gracias al cual se controlan los fallos por caída de los nodos primario y copia, así como la promoción de nodos si tienen lugar dichas caídas.



Validación

Para comprobar que todas las decisiones de diseño eran acertadas han sido de gran utilidad las funciones de prueba proporcionadas ya que han agilizado el proceso de detección y corrección de errores en el código además de conseguir una mayor familiarización con el método de verificar aserciones. Ej. (assert p ==: undefined)

En gran medida las pruebas se hacen verificando las vistas, tanto la vista válida como la tentativa dependiendo de la prueba en cuestión. De las vistas se observa el número de vista, así como los nodos primario y copia.

En muchas ocasiones durante el proceso de diseño al no ser suficiente la información proporcionada por la función de prueba ha sido necesario realizar diversas impresiones repartidas por el código para verificar la vista en cada función, seguir su recorrido y de esta manera sacar las conclusiones oportunas acerca de donde podría estar produciéndose un fallo ya que la vista no coincide con las expectativas en una fase determinada.

Para abordar las pruebas 7, 8 y 9 ha sido necesario realizar funciones de prueba propias que recuerdan bastante a las originalmente proporcionadas. Se construye un escenario inicial a partir del cual pasarán diferentes acciones con los nodos, una vez terminadas se comprueba que las expectativas sobre la vista coinciden con las devueltas tras la ejecución del código.

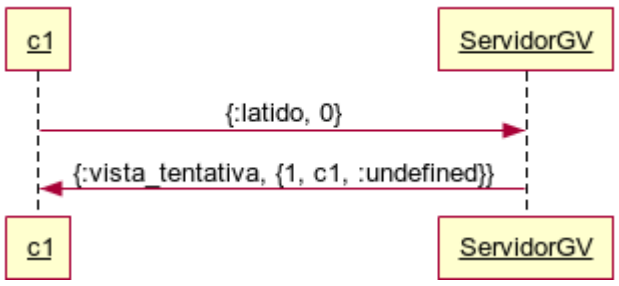
A continuación, se describen las pruebas que se han pasado para comprobar el completo y correcto funcionamiento de muchos casos del servidor de vistas

Test 1

En el primer test se comprueba que antes de que el servidor reciba algún latido de un nodo no haya ningún nodo en la vista como primario.

De esta manera, si el valor de primario es igual a :undefined se considera que el test se ha pasado correctamente.

Test 2



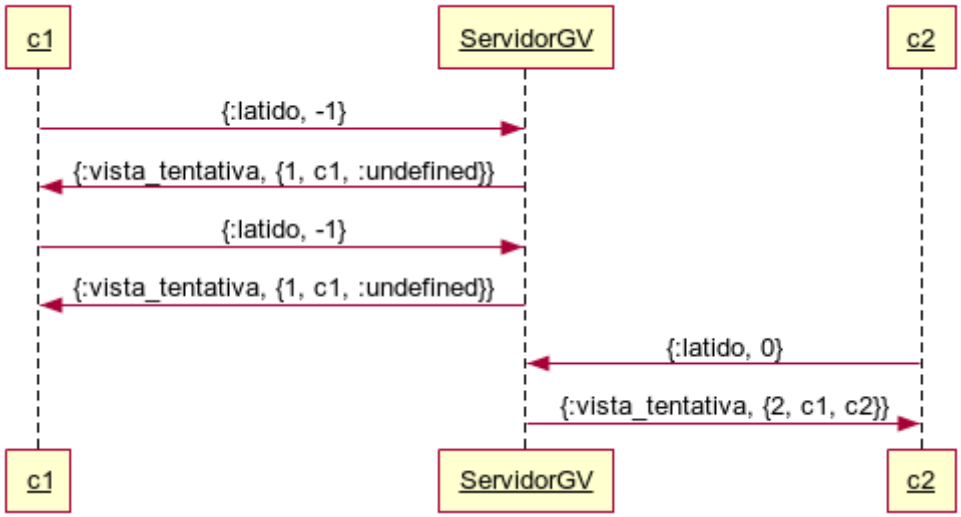
Intercambio de mensajes en la prueba 2

En esta prueba se verifica que cuando un nodo envía un latido , el servidor debería responder añadiéndolo como nodo primario en la vista tentativa.

De esta manera, se envía desde c1 un latido con un 0 al servidor Gestor de Vistas, el cual trata el mensaje y añade al nodo c1 como primario en la vista.

Finalmente se comprueba que los cambios figuren en la vista tentativa.

Test 3



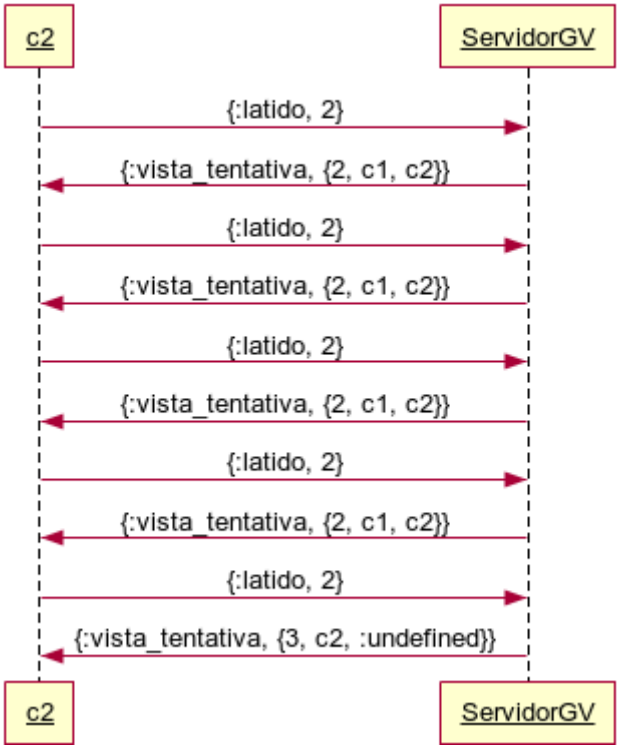
Intercambio de mensajes en la prueba 3

En este caso se va a añadir un segundo nodo, el cual, como ya existe un nodo primario, debería ser añadido como nodo copia en la vista tentativa.

Al comienzo se envía un latido con un -1 desde el nodo c1 con la finalidad de que la vista no se confirme aún como válida y posteriormente el nodo c2 se comunica con el servidor con un latido con un 0, avisando que ya se encuentra disponible. De esta manera, el servidor debería añadir este nodo como copia a la vista tentativa.

Finalmente, al igual que en la prueba anterior, se verifican los cambios mirando la vista tentativa.

Test 4



Intercambio de mensajes en la prueba 4

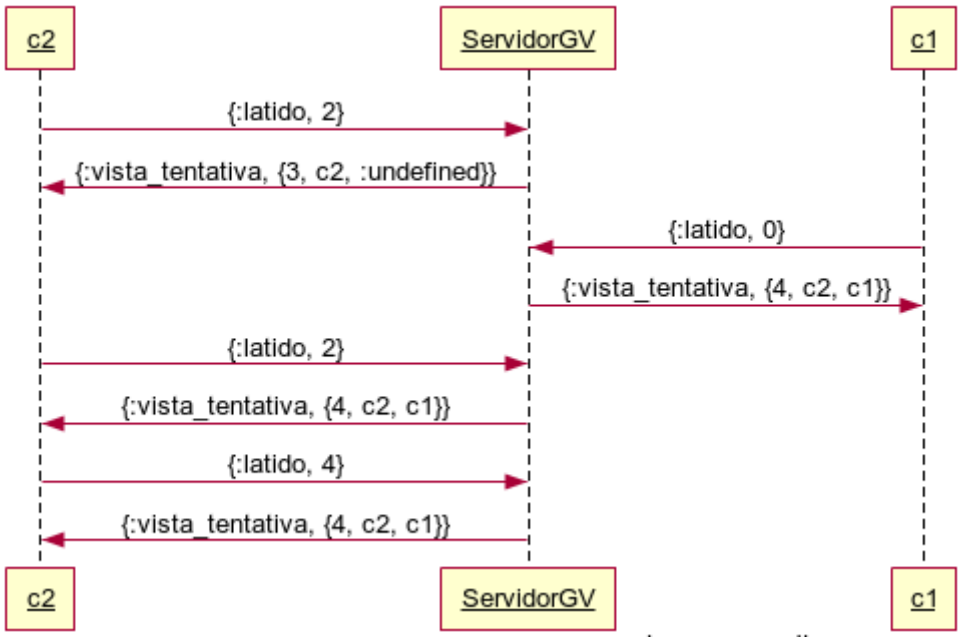
En la siguiente prueba se quiere probar que el servidor del Gestor de Vistas maneja correctamente las caídas de los nodos cuando no se reciben un determinado número de latidos.

Para simular este comportamiento se hace una llamada a una función que, cada 50ms, hace que el nodo c2 envíe un latido al servidor mientras que el nodo c1 no hace nada.

Por lo tanto, tras 4 latidos del nodo c2 y 4 latidos fallidos por parte del nodo c1, el servidor debería tratar al nodo c1 como caído y eliminarlo de la vista.

En este caso también se verifica el correcto resultado mediante la comprobación de la vista tentativa.

Test 5



Intercambio de mensajes en la prueba 5

En la quinta prueba se quiere comprobar que el nodo que había caído en el caso anterior se incluye como copia en el momento en el que rearranque (latido con un 0).

Para ello se comienza con un latido por parte del nodo c2 con la finalidad de obtener la vista tentativa que se encuentra en el servidor. Entonces, se envía un latido por parte del nodo c1 con valor 0 para indicar que ha arrancado y que ya se vuelve a encontrar operativo.

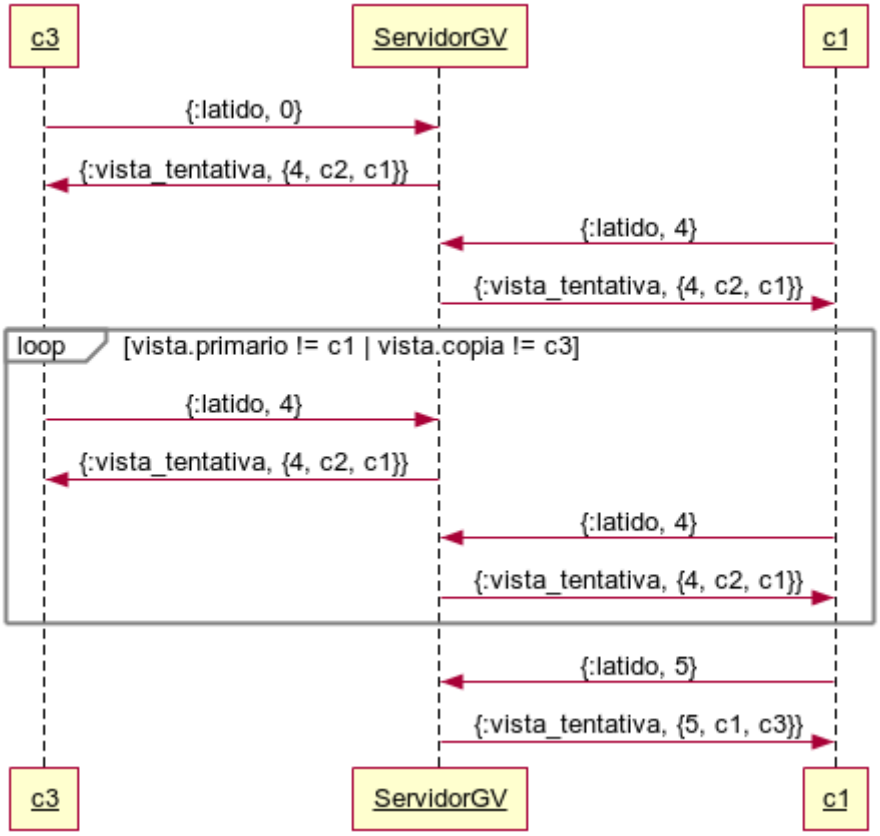
Finalmente se intercambian mensajes entre el servidor y el nodo c2 hasta que se confirma la nueva vista tentativa, con número de vista igual a 4, como vista válida.

En esta prueba se comprueba la validez mediante la vista válida.

Test 6

En la siguiente prueba se va a comprobar que cuando el nodo primario falla, se promociona correctamente el nodo copia a primario y el primer nodo de la lista a de nodos en espera a copia.

Para ello se envía al inicio un latido con un 0 desde el nodo c3 con la finalidad de que sea incluido como nodo en espera para luego ser promocionado a copia cuando falle el primario.

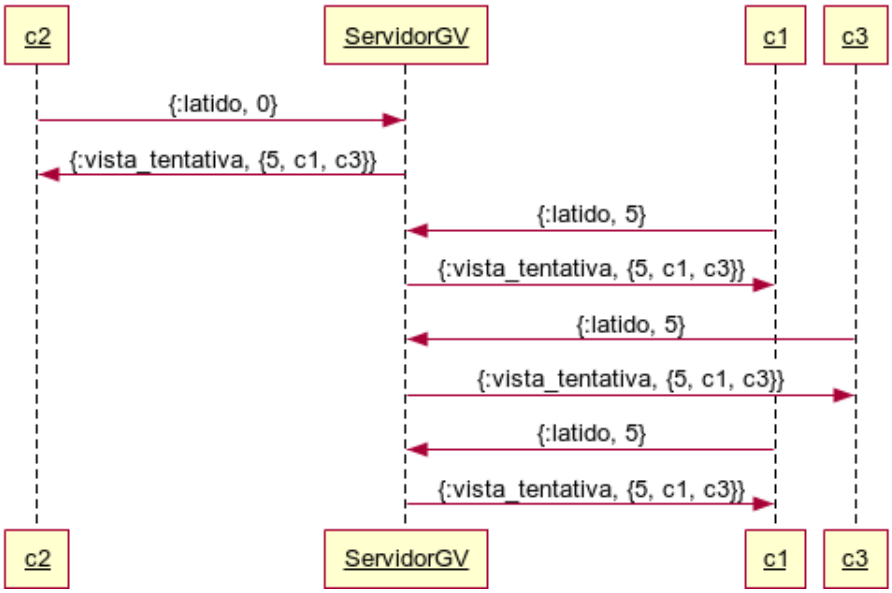


Intercambio de mensajes en la prueba 6

Para simular el fallo del primario se ejecuta un bucle en el que, en cada iteración, los nodos c3 y c1 enviarán latidos al servidor mientras que el nodo c2 no hará nada para que el gestor lo considere caído.

Tras la ejecución de dicho bucle se comprueba c1, que debería ser el nuevo primario, envía un latido para confirmar la vista tentativa como válida y finalmente se verifica la prueba con los valores de la vista válida.

Test 7



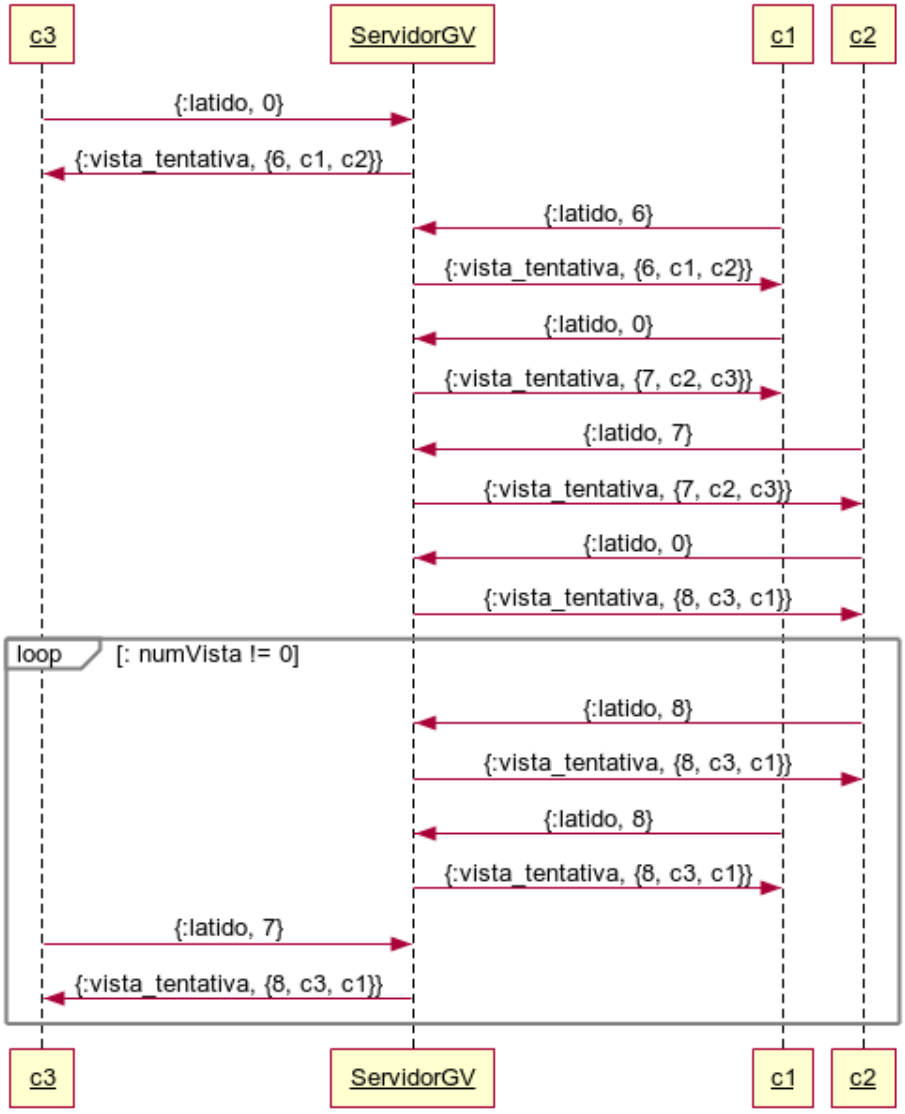
Intercambio de mensajes en la prueba 7

El objetivo de esta prueba es verificar que si el nodo que se consideraba caído en el caso anterior vuelve a estar operativo se incluya dicho nodo en la lista de espera.

Por lo tanto, inicialmente se envía un latido con un 0 desde el nodo c2, que es el que había caído anteriormente, para avisar de su disponibilidad y posteriormente se intercambian latidos entre el resto de los nodos y el servidor.

Si el gestor actúa correctamente se puede comprobar que ni la vista tentativa ni la válida se ven afectadas por la inclusión de un nuevo nodo en la lista de espera.

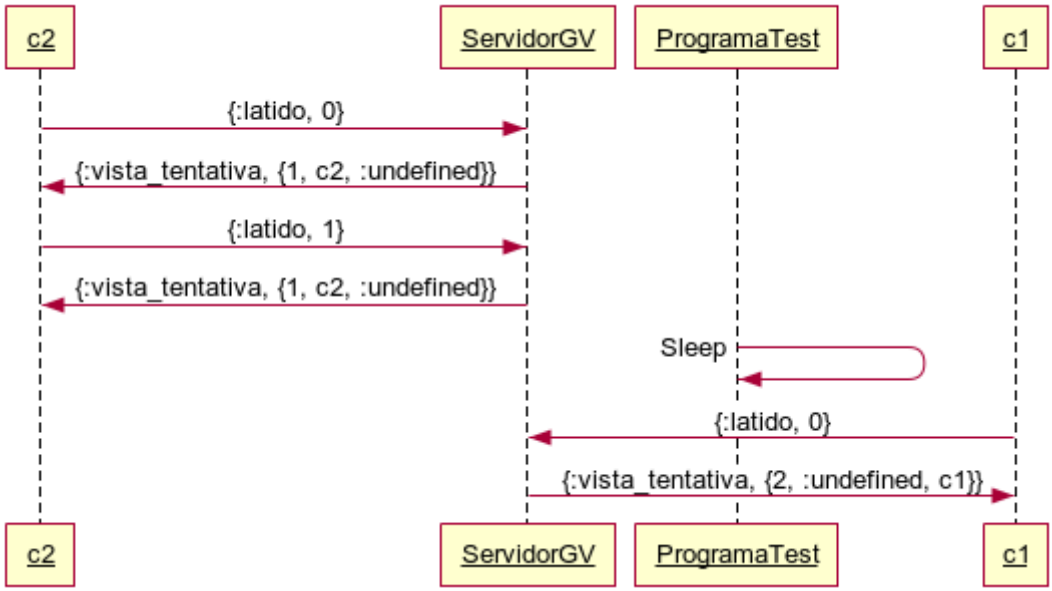
Test 8



En esta prueba se pretende probar que el gestor de vistas falla cuando se llega a una situación de error crítico. Para ello se genera un escenario en el cual el primario cae antes de haber confirmado como válida la vista en la que figuraba como primario.

En el servidor se ha simulado la parada por fallo crítico como un reinicio, de manera que, si se ha dado dicho error, las vistas tentativa y válida del gestor serán iguales a las resultantes de la ejecución de la función vista_inicial().

Test 9:



Intercambio de mensajes en la prueba 9

En esta última prueba se quiere comprobar que cuando se tiene una vista sin primario ni copia que no sea la vista inicial no se puede colocar directamente como primario a un nodo cualquiera sin que previamente fuera copia.

Para ello se recrea la situación descrita, comprobando al final que el valor de primario en la vista tentativa sea :undefined.

Conclusión

Tras finalizar esta práctica hemos deducido lo siguiente:

Un sistema distribuido tolerante a fallos con estado puede resultar muy complejo de analizar si cuenta con un número de nodos primario/copia considerable, ya que si suelen "caerse" con regularidad la promoción entre nodos en espera y copia puede ser excesiva.

El sistema resulta muy tolerante a fallos con los nodos, pero el gestor de vistas se encuentra realizando una misión muy crítica ya que es el que más responsabilidad presenta referente a la continuidad del sistema y si se produce un fallo en éste puede ser verdaderamente grave.

Este sistema es más interesante que la detección a fallos sin estado con menor redundancia y mejores prestaciones, aunque aún puede presentar fallos lo que no lo hace inmune ni perfecto.