Servicio de almacenamiento basado en primario/copia

Sistemas distribuidos - Práctica 5

Óscar Baselga Lahoz, Alberto Calvo Rubió
Universidad de Zaragoza

29 de diciembre de 2019

Índice

1. Introducción

En esta práctica se plantea desarrollar un servicio de almacenamiento distribuido clave/valor en memoria RAM, tolerante a fallos, utilizando el servicio de vistas implementado en la Práctica 4. Las operaciones que se pondrán a disposición de los clientes serán: lectura de valor y escritura de valor. Los protocolos a diseñar son los correspondientes al esquema de funcionamiento Primario/Copia planteado como ejemplo en la clase de teoría. Se plantearan situaciones de fallo adicionales, como por ejemplo, el envío de escrituras duplicadas o la pérdida de mensajes.

2. Servidor de almacenamiento

2.1. Datos

En esta ocasión, el servidor a diseñar, tendrá que mantener unos datos a modo de base de datos, además de los necesarios para llevar a cabo toda la lógica del sistema. El struct que almacena consiste en un Map con los siguientes elementos:

- **vista**: la vista actual enviada por el servidor gestor de vistas. Con ella se puede conocer si el propio servidor ha sido promocionado a primario, copia o si se ha decretado como servidor en espera (el gv puede haberlo detectado como caído).
- **datos**: la base de datos a mantener. También de tipo Map.
- **peticiones**: Map que almacena el identificador de la última petición de cada cliente, de forma que puede identificar peticiones repetidas o antiguas y de esta forma no llevarlas a cabo.
- **automata**: indica el estado en el que se encuentra el servidor (:primario, :copia, :proxima_copia, :esperando_inicializacion_copia, :esperando_confirmacion_copia, :nodo_tipo_espera).

2.2. Comportamiento

Para definir el comportamiento de cada uno de los servidores del servicio de almacenamiento, se ha realizado un diagrama de estados que representa la posición de cada uno de ellos.

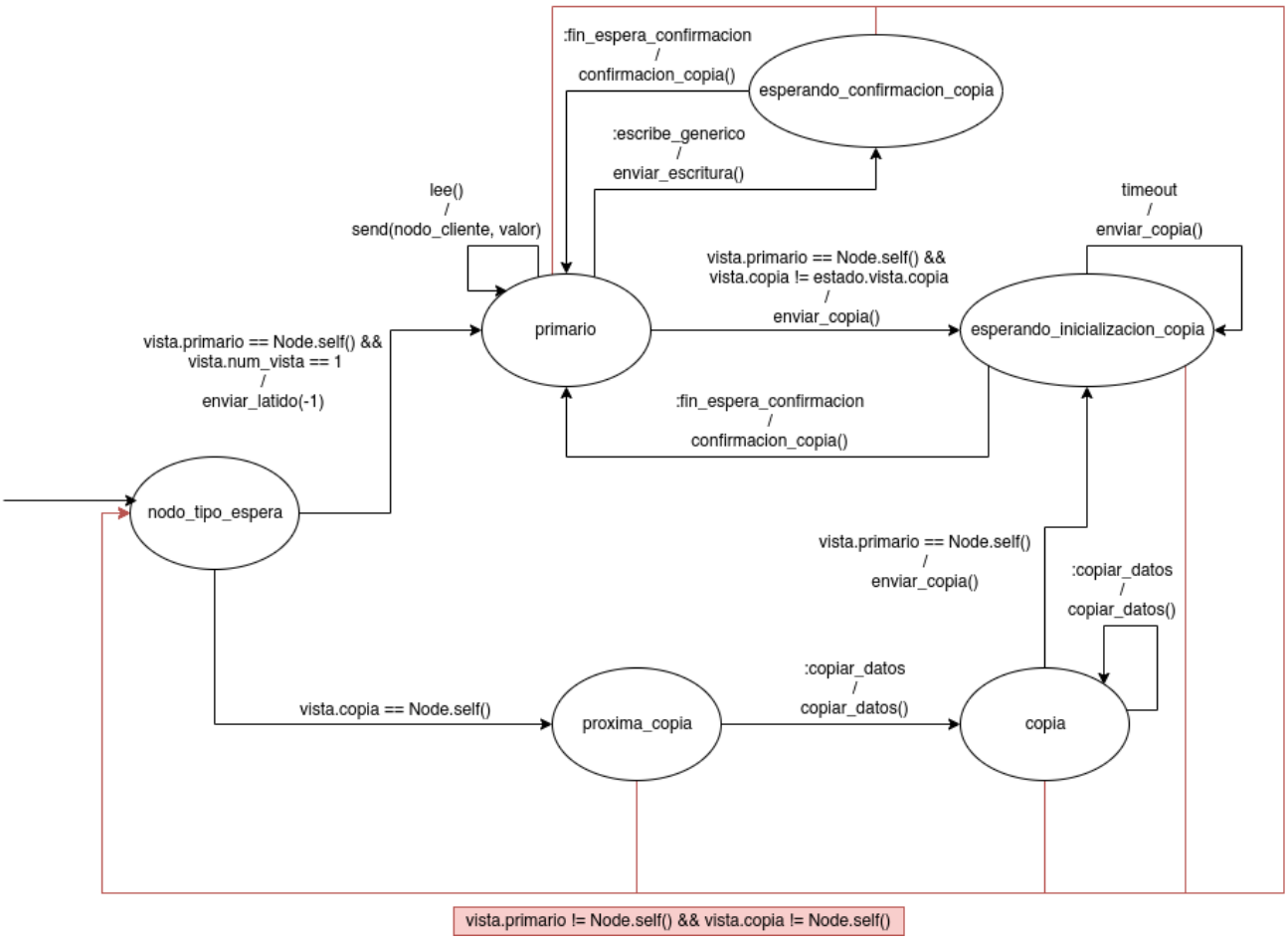

Figura 1: Test 4

- **nodo_tipo_espera**: estado inicial de todo servidor de almacenamiento. Puede convertirse en primario si el gv lo establece como tal y el número de vista es 1. También puede convertirse en proxima_copia si la nueva vista así lo indica. Desde todos los estados se puede volver a este si lo indica el gv.
- **primario**: recibe las peticiones de los clientes. Las lecturas se responden directamente, en cambio, las escrituras se propagan a la copia(de forma recursiva hasta que confirma) y se pasa al estado de esperando_confirmacion_copia. Además puede detectar que se ha cambiado de nodo_copia y por tanto le envía los datos y se pasa al estado de esperando_inicializacion_copia.
- **esperando_inicializacion_copia**: en este estado se permite la respuesta de lecturas. Cuando llega la confirmación de la copia, se vuelve al estado primario.

- **esperando_confirmacion_copia**: se espera a que la copia confirme una nueva escritura y entonces se responderá al cliente y se pasará al estado primario.
- **proxima_copia**: el nodo espera a recibir los datos del primario. Una vez copiados, se pasará al estado copia.
- **copia**: recibe solicitudes de copia de datos por el servidor primario y además puede ser promocionado a primario en caso de indicarlo la vista enviada por el gv, en cuyo caso pasará al estado esperando_inicializacion_copia a la espera de que confirme la nueva copia a la que haya enviado los datos.

3. Validación

Para comprobar el correcto funcionamiento del sistema se han realizado diversos test que simulan situaciones delicadas. En todos ellos se inicializan los servidores y se comprueba un correcto inicio.

3.1. Test 4

En este test se pretende comprobar la concurrencia y que se mantienen los datos tras la caída del primario y de la copia (una vez haya sustituido al anterior primario y realizado la nueva copia). Se leerá los datos del tercer servidor para asegurar la consistencia.

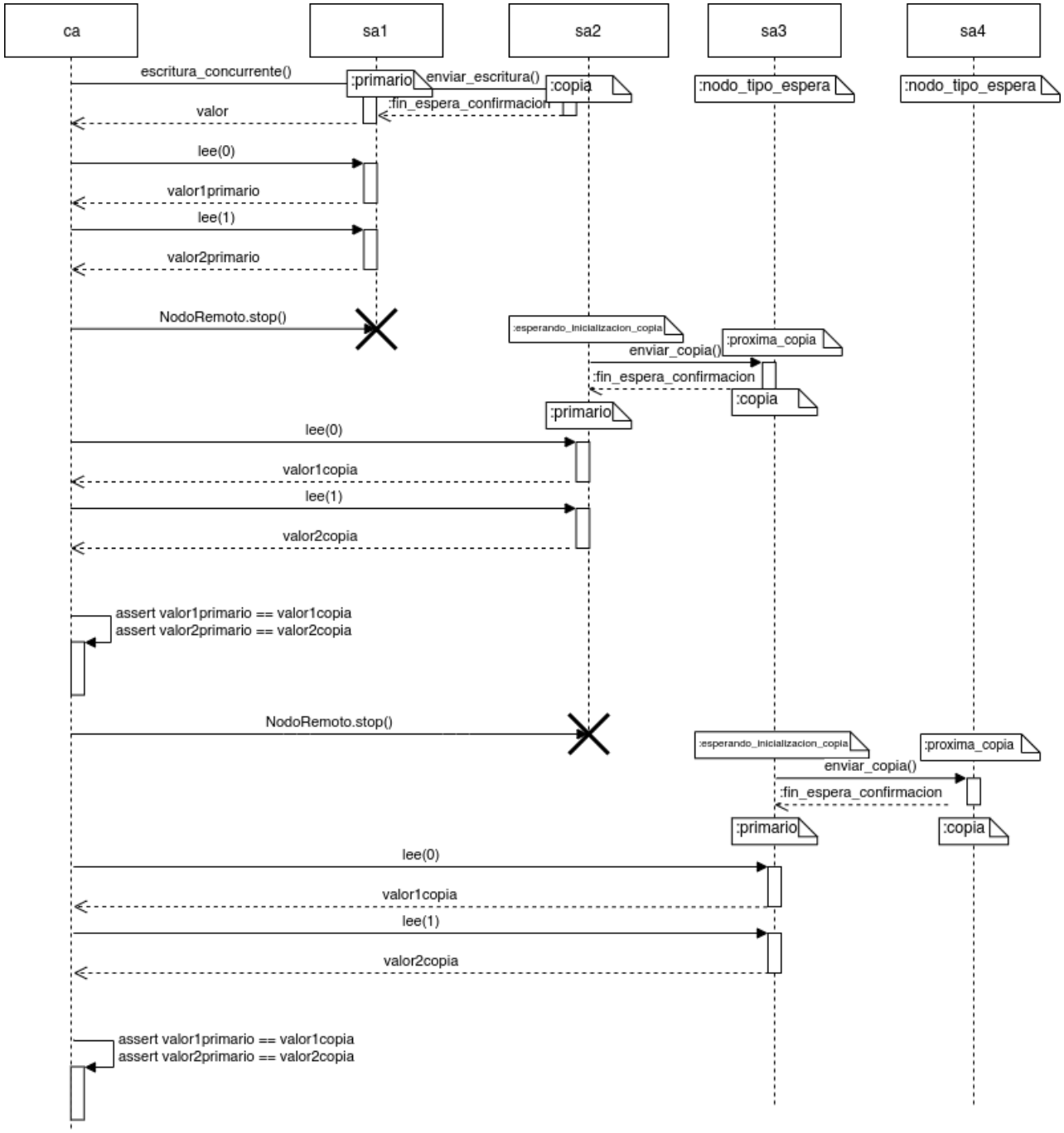

Figura 2: Test 4

3.2. Test 5

Tras caer la copia, se intenta realizar una escritura inmediata al servidor primario. Al no disponer de ninguna copia, este intenta inicializar una. Mientras tanto, no responde a la petición de escritura (se prioriza **consistencia** frente a disponibilidad). Una vez haya copia, se procesará la petición (con su respectiva propagación). A continuación, se para el primario y se comprueba la consistencia en el tercer servidor (en este momento tercer primario)

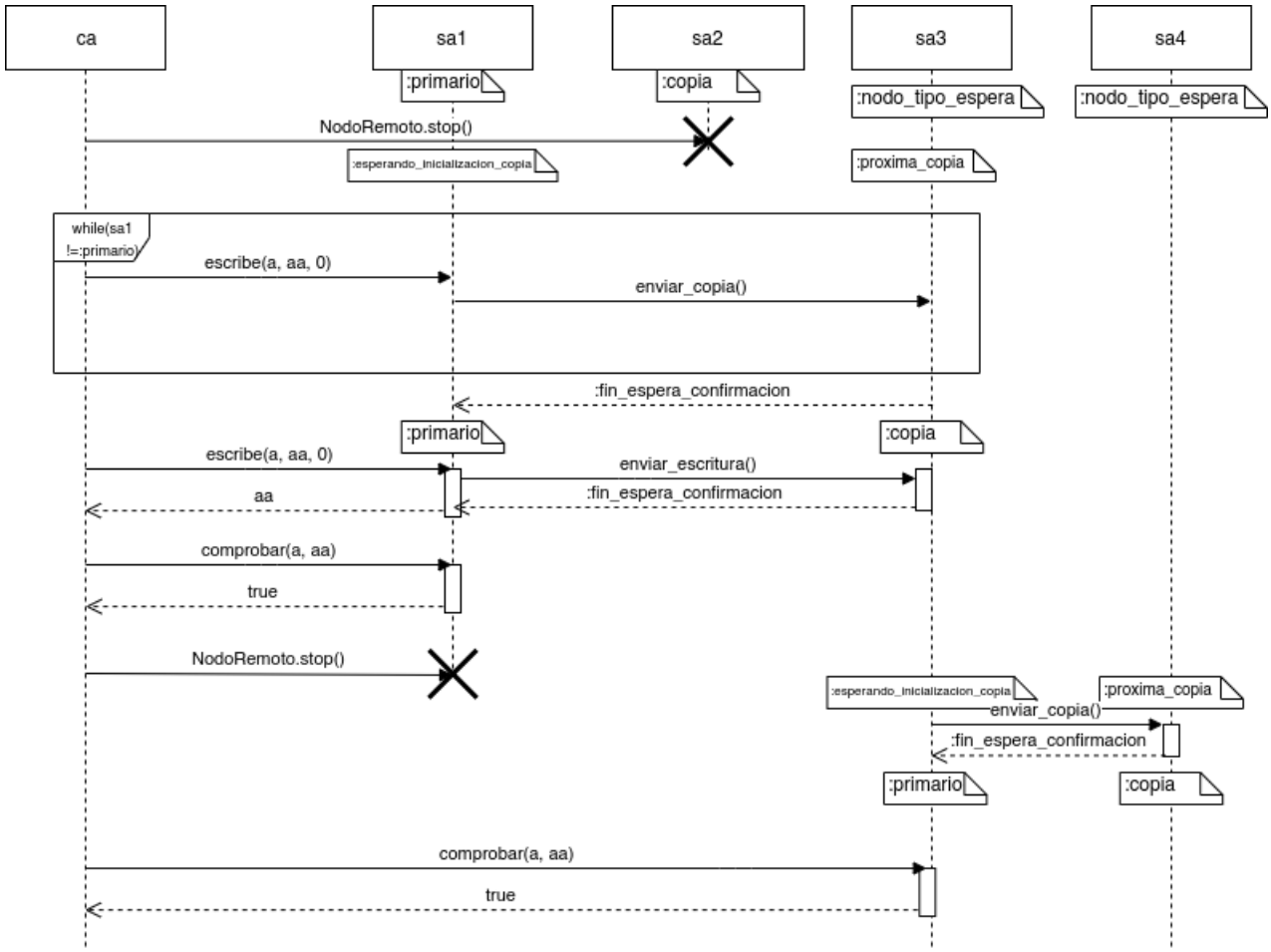

Figura 3: Test 5

3.3. Test 6

En este test se realiza una escritura duplicada, representada con el mismo identificador. Primero se realiza la escritura por el ca1. Después escribe el segundo servidor cliente, ca2, un valor diferen-

te. Finalmente, escribe ca1 simulando que no ha recibido la respuesta del primario, repitiendo la petición. Al ser la misma, sa1 no debe propagar la escritura y debe mantener el valor que había escrito ca2.

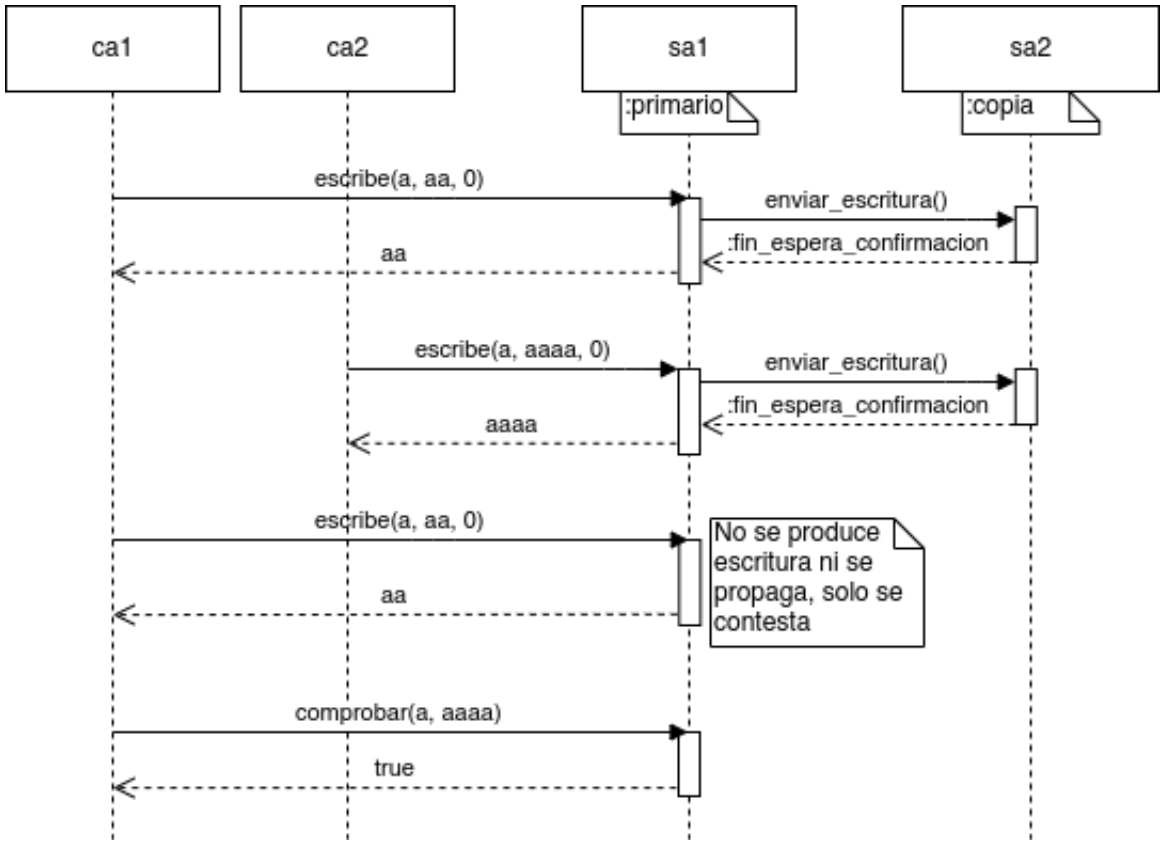

Figura 4: Test 6

3.4. Test 7

Tras realizar una escritura en el primer primario, se simula su caída enviando un latido(0) pero manteniendo los datos del servidor, ya que no ha caído realmente. Se hace una petición al antiguo primario después de que se reestructure el sistema, pero este contesta :no_soy_primario_valido asegurando que un antiguo primario no debe responder peticiones.

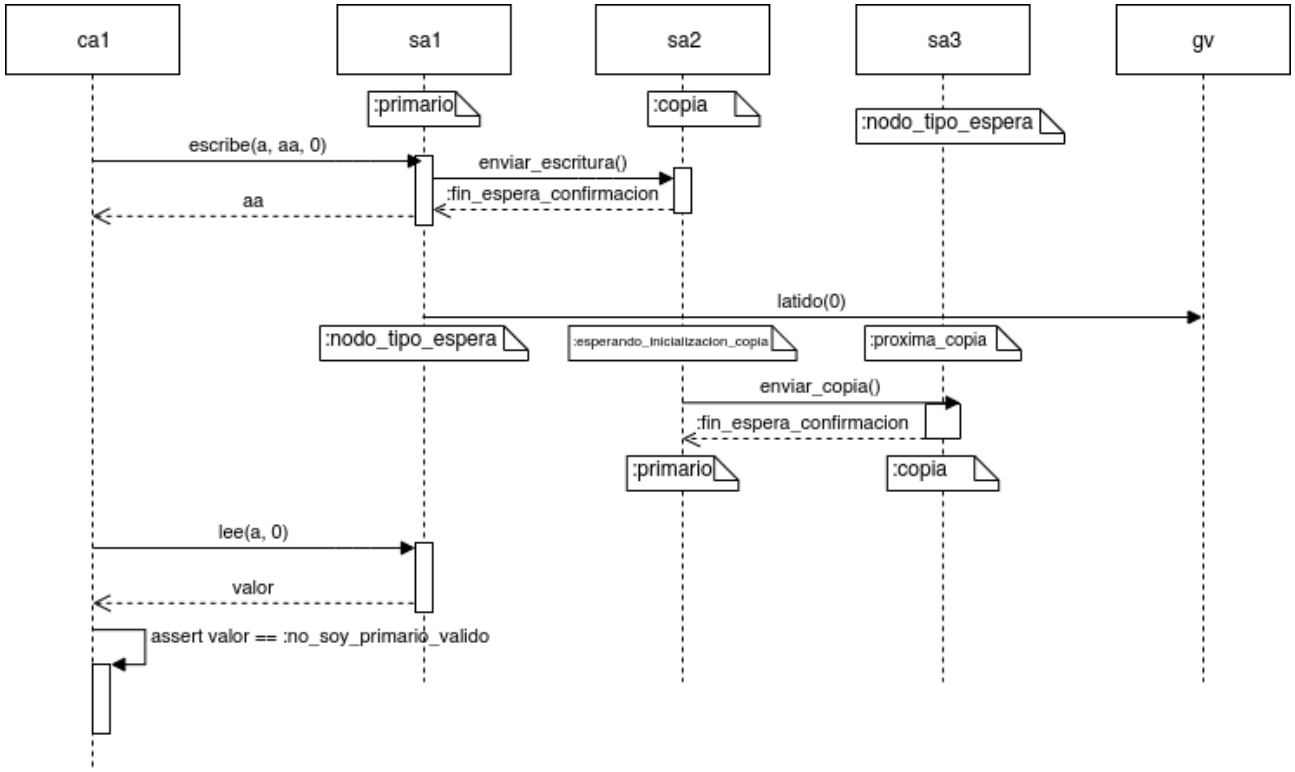

Figura 5: Test 7