(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 774 691**

(51) Int. CI.:

H04W 4/00 (2008.01)
H04L 29/08 (2006.01)
H04L 12/18 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **04.10.2013 PCT/US2013/063454**

(87) Fecha y número de publicación internacional: **03.07.2014 WO14105248**

(96) Fecha de presentación y número de la solicitud europea: **04.10.2013 E 13867648 (1)**

(97) Fecha y número de publicación de la concesión europea: **12.02.2020 EP 2939446**

(54) Título: **Unión de afiliaciones en sistemas informáticos distribuidos**

(30) Prioridad:

28.12.2012 US 201261746940 P
15.03.2013 US 201313837366

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
22.07.2020

(73) Titular/es:

WANDISCO, INC. (100.0%)
Suite 270, Bishop Ranch 8, 5000 Executive Parkway
San Ramon, CA 94583, US

(72) Inventor/es:

AAHLAD, YETURU;
PARKIN, MICHAEL y
AKHTAR, NAEEM

(74) Agente/Representante:

VALLEJO LÓPEZ, Juan Pedro

DESCRIPCIÓN

Unión de afiliaciones en sistemas informáticos distribuidos

Antecedentes

Los proyectos colaborativos, que a menudo se facilitan de una manera concurrente entre recursos globalmente separados (es decir, proyectos colaborativos de múltiples sitios), se han vuelto comunes para cualquier número de diferentes tipos de proyectos. Ejemplos de tales proyectos incluyen, pero sin limitación, desarrollo de software, diseño de aviones y diseño de automóviles. Basarse en los recursos distribuidos (por ejemplo, recursos en ubicaciones físicamente diferentes, ubicaciones lógicamente diferentes, etc.) para acelerar las líneas de tiempo de proyecto a través de la optimización de utilización de recursos humanos y aprovechamiento de las habilidades de recursos globales ha demostrado por sí mismo que ofrece resultados ventajosos.

Una solución informática distribuida usada al facilitar un proyecto colaborativo de múltiples sitios se denomina en el presente documento como una solución informática colaborativa de múltiples sitios distribuida. Sin embargo, una solución informática colaborativa de múltiples sitios distribuida es únicamente un ejemplo de una solución informática distribuida. En un ejemplo, una solución informática distribuida comprende una red de ordenadores que operan un automóvil. En otro ejemplo, una solución informática distribuida comprende una red de ordenadores en una ubicación geográfica (un centro de datos). En otro ejemplo más, una solución informática distribuida es una pluralidad de ordenadores conectados a un encaminador (es decir, una subred).

Aunque existen las soluciones informáticas distribuidas convencionales, no existen sin limitaciones que impactan de manera adversa su efectividad, fiabilidad, disponibilidad, escalabilidad, transparencia y/o seguridad. En particular, con respecto a soluciones informáticas colaborativas de múltiples sitios distribuidas convencionales están limitadas en su capacidad para sincronizar trabajo de sitios de desarrollo distribuidos globalmente en una manera tolerante a fallos en tiempo real. Esta incapacidad fuerza cambios en el desarrollo de software y procedimientos de entrega que a menudo provocan retardos y aumentan riesgos. Por consiguiente, no se consiguen completamente ahorros en costes y mejoras en la productividad que deberían realizarse a partir de la implementación de un proyecto colaborativo que utiliza una solución informática distribuida convencional.

Las soluciones informáticas colaborativas de múltiples sitios distribuidas convencionales fuerzan de manera indeseable a que los usuarios cambien sus procedimientos de desarrollo. Por ejemplo, las soluciones informáticas colaborativas de múltiples sitios distribuidas convencionales que carecen de funcionalidades ventajosas asociadas a capacidades de gestión de información en tiempo real tienen un problema fundamental en que no pueden garantizar que los repositorios de Sistemas de Versiones Concurrentes (CVS) estarán en sincronización en cualquier punto en el tiempo. Esto significa que hay una enorme probabilidad de que los desarrolladores en diferentes sitios puedan sobrescribir o corromper de manera inadvertida el trabajo unos de los otros. Para evitar tal sobrescrita y corrupción potencial, estas soluciones informáticas colaborativas de múltiples sitios distribuidas convencionales requieren una ramificación de código fuente excesiva y/o propensa a errores y unión de ficheros manual para formar parte del proceso de desarrollo. Esto fuerza de manera eficaz a que se particione el trabajo de desarrollo basándose en zonas de tiempo y hace extremadamente desafiante, si no imposible, la colaboración entre equipos de desarrollo distribuidos.

Una máquina de estado replicada es un habilitador preferido de soluciones informáticas distribuidas. Uno de varios ejemplos posibles de una solución informática distribuida es un repositorio de información replicado. Por lo tanto, más particularmente, una máquina de estado replicada es un habilitador preferido de repositorios de información replicados. Una de varias aplicaciones posibles de repositorios de información replicados son las soluciones informáticas colaborativas de múltiples sitios distribuidas. Por lo tanto, más particularmente, una máquina de estado replicada es un habilitador preferido de soluciones informáticas colaborativas de múltiples sitios distribuidas.

Por consiguiente, las soluciones informáticas distribuidas a menudo se basan en máquinas de estado replicadas, repositorios de información replicados o ambos. Las máquinas de estado replicadas y/o los repositorios de información replicados proporcionan la generación, manipulación y gestión concurrente de información y, por lo tanto, son aspectos importantes de la mayoría de las soluciones informáticas distribuidas. Sin embargo, los enfoques conocidos para facilitar la replicación de máquinas de estado y facilitar la replicación de repositorios de información no están exentos de sus desventajas.

Las implementaciones convencionales para facilitar la replicación de máquinas de estado tienen una o más desventajas en que limitan su efectividad. Una de tales desventajas es que son propensas a la anticipación repetida de los proponentes en un protocolo de acuerdo, que impacta de manera adversa a la escalabilidad. Otra desventaja de este tipo es que la implementación de la optimización de líder débil requiere la elección de un líder, que contribuye a que tal optimización impacte de manera adversa a la complejidad, velocidad y escalabilidad, y requiera uno o más mensajes por acuerdo (por ejemplo, 4 en lugar de 3), que impacta de manera adversa a la velocidad y escalabilidad. Otra desventaja de este tipo es que los acuerdos tienen que alcanzarse secuencialmente, que impacta de manera adversa a la velocidad y escalabilidad. Otra desventaja de este tipo es que la recuperación de

almacenamiento persistente está limitada, si no completamente ausente, lo que impone una carga considerable en el desarrollo puesto que el almacenamiento necesita un despliegue de este tipo que crecerá continuamente y, potencialmente, sin límite. Otra desventaja de este tipo es que el manejo eficaz de grandes propuestas y grandes números de pequeñas propuestas está limitado, si no completamente ausente, que afecta de manera adversa la escalabilidad. Otra desventaja de este tipo es que debe comunicarse un número relativamente alto de mensajes para facilitar la replicación de máquina de estado, que afecta de manera adversa a la escalabilidad y a la compatibilidad de red de área extensa. Otra limitación es que los retardos al comunicar mensajes impactan de manera adversa la escalabilidad. Otra desventaja de este tipo es que tratar escenarios de fallo cambiando dinámicamente (por ejemplo, incluyendo y excluyendo según sea necesario) participantes en la máquina de estado replicada impacta de manera adversa a la complejidad y escalabilidad.

Las implementaciones convencionales para facilitar la replicación de repositorios de información tienen una o más desventajas que limitan su efectividad. Una de tales desventajas es que ciertas soluciones informáticas colaborativas de múltiples sitios convencionales requieren un único coordinador central para facilitar la replicación de repositorios de información centralmente coordinados. De manera indeseable, el coordinador central afecta de manera adversa a la escalabilidad puesto que todas las actualizaciones al repositorio de información deben encaminarse a través del único coordinador central. Adicionalmente, una implementación de este tipo no está altamente disponible debido a que el fallo del coordinador central único provocará que la implementación deje de poder actualizar cualquier réplica del repositorio de información. Otra desventaja de este tipo es que, en una implementación de replicación de repositorio de información que se basa en reproducciones de registro, la replicación de repositorio de información se facilita de una manera activa-pasiva. Por lo tanto, únicamente puede actualizarse una de las réplicas en cualquier momento dado. Debido a esto, la utilización de recursos es pobre debido a que otras réplicas están en espera o limitadas a servir una aplicación de solo lectura, tal como, por ejemplo, una aplicación de minería de datos. Otra desventaja de este tipo resulta cuando la implementación se basa en replicación débilmente consistente respaldada por heurística de resolución de conflicto y/o mecanismos de intervención de aplicación. Este tipo de replicación de repositorio de información permite actualizaciones que entran en conflicto con las réplicas del repositorio de información y requieren una aplicación que usa el repositorio de información para resolver estos conflictos. Por lo tanto, una implementación de este tipo afecta de manera adversa a la transparencia con respecto a la aplicación.

Haciendo referencia aún a implementaciones convencionales de facilitación de la replicación de repositorios de información, tienen aún una o más desventajas que limitan su efectividad, las implementaciones que están basadas en una solución de espejado de disco se conoce que tienen una o más desventajas. Este tipo de implementación es una implementación activa-pasiva. Por lo tanto, una de tales desventajas es que únicamente puede usarse una de las réplicas por la aplicación en cualquier momento dado. Debido a esto, la utilización de recursos es pobre debido a que las otras réplicas (es decir, los espejos pasivos) ni son legibles ni escribibles mientras están en su papel como espejos pasivos. Otra desventaja de este tipo de esta implementación particular es que el método de replicación no tiene conocimiento de los límites de transacción de la aplicación. Debido a esto, en el punto de un fallo, el espejo puede tener un resultado parcial de una transacción y, por lo tanto, puede ser inutilizable. Otra desventaja de este tipo es que el método de replicación propaga cambios a la información del nodo en el que se origina el cambio a todos los otros nodos. Debido a que el tamaño de los cambios a la información es a menudo mucho mayor que el tamaño del comando que provocó el cambio, una implementación de este tipo puede requerir una cantidad indeseablemente grande de ancho de banda. Otra desventaja de este tipo es que, si la información en el repositorio maestro se corrompiera por cualquier razón, esa corrupción se propagaría a todas las otras réplicas del repositorio. Debido a esto, el repositorio de información puede no recuperarse o puede tener que recuperarse desde una copia de respaldo más antigua, conllevando de esta manera a pérdida de información adicional.

El documento US 7069320-Chang et al describe una red que tiene una pluralidad de nodos que está reconfigurada para reflejar un cambio en la topología de la red. En particular, tras recibir una solicitud de reconfiguración, cada nodo entra en un estado quiescente durante un periodo de tiempo predeterminado suficiente para permitir que al menos otro nodo entre también en un estado quiescente. A continuación, tras la terminación del estado quiescente, el nodo está reconfigurado para reflejar el cambio en la topología de la red sin tener que comprobar con cualesquiera otros nodos de la red. En otras realizaciones, el periodo predeterminado de tiempo es suficiente para permitir que los protocolos actualmente en ejecución completen la ejecución, así como para permitir la transmisión de solicitudes de reconfiguración para propagar la reconfiguración en la red.

El documento US 6247059-Johnson et al se refiere a un sistema de procesamiento múltiple que comprende múltiples nodos comunicativamente interconectados, teniendo cada nodo una o más unidades de procesador, los mensajes de multidifusión envidos por un nodo emisor contendrán información que permite que los nodos receptores pretendidos comprueben y determinen la posibilidad de que no se recibieran mensajes de multidifusión enviados más anteriores desde el nodo emisor por el nodo receptor.

Sumario de la invención

De acuerdo con un primer aspecto de la presente invención, se proporciona un método implementado por ordenador de despliegue de una afiliación de nodos en un sistema informático distribuido, el método implementado por ordenador de acuerdo con la reivindicación 1.

En un segundo aspecto de la invención, se proporciona un dispositivo informático de acuerdo con la reivindicación 7.

En un aspecto adicional de la invención, se proporciona un medio legible por máquina tangible de acuerdo con la reivindicación 13.

Por lo tanto, sería útil y ventajosa una máquina de estado replicada que superara las desventajas asociadas a máquinas de estado replicadas convencionales. Más específicamente, un repositorio de información replicada creado usando una máquina de estado replicada de este tipo sería superior a un repositorio de información replicada convencional. Incluso más específicamente, un repositorio de CVS replicado creado usando una máquina de estado replicada de este tipo sería superior a un repositorio de CVS replicado convencional.

Formar una afiliación, o una colección específica de entidades a partir de un conjunto de entidades conocidas, es útil en sistemas informáticos distribuidos tales como se han descrito anteriormente, de modo que la información pueda compartirse entre agrupaciones especificadas de nodos confiables.

Breve descripción de los dibujos

La Figura 1 es un diagrama de bloques que muestra relaciones funcionales de elementos dentro de una arquitectura de sistema informático de múltiples sitios.

La Figura 2 es un diagrama de bloques de alto nivel que muestra un despliegue de elementos que componen una arquitectura de sistema informático de múltiples sitios.

La Figura 3 es un diagrama de bloques que muestra componentes funcionales de una máquina de estado replicada.

La Figura 4 es un diagrama de bloques que muestra una propuesta emitida por un nodo de aplicación local.

La Figura 5 es un diagrama de bloques que muestra la estructura de entrada de un secuenciador global de la máquina de estado replicada de la Figura 3.

La Figura 6 es un diagrama de bloques que muestra la estructura de entrada de un secuenciador local de la máquina de estado replicada de la Figura 3.

La Figura 7 es un diagrama de bloques que muestra un replicador.

La Figura 8 es un diagrama de bloques de nivel detallado que muestra un despliegue de elementos que componen una arquitectura de sistema informático de múltiples sitios.

La Figura 9 muestra aspectos de un método de despliegue de una afiliación de nodos en un sistema informático distribuido, de acuerdo con una realización.

La Figura 10 muestra aspectos adicionales de un método de despliegue de una afiliación de nodos en un sistema informático distribuido, de acuerdo con una realización.

La Figura 11 es un diagrama de bloques de un dispositivo informático con el que pueden implementarse las realizaciones.

Descripción detallada

Se desvelan en el presente documento diversos aspectos para facilitar una implementación práctica de una máquina de estado replicada en diversas arquitecturas de sistema informático distribuido (por ejemplo, arquitectura de sistema informático colaborativo de múltiples sitios distribuida). Un experto en la materia estará al tanto de una o más implementaciones convencionales de una máquina de estado replicada. Por ejemplo, una implementación convencional de este tipo de una máquina de estado se desvela en la publicación titulada "Implementing fault-tolerant services using the state machine approach: A tutorial" (páginas 299-319), de autoría de F. B. Schneider, publicada en ACM Computing Surveys 22 en diciembre de 1990. Con respecto a la implementación convencional de una máquina de estado en una arquitectura de sistema de aplicación distribuido y como se analiza a continuación en mayor detalle, las realizaciones potencian aspectos de escalabilidad, fiabilidad, disponibilidad y tolerancia a fallos.

El sistema y método asociado descritos en el presente documento proporcionan una implementación práctica de una máquina de estado replicada en diversas arquitecturas de sistema informático distribuido (por ejemplo, arquitecturas de sistema informático colaborativo de múltiples sitios distribuida). Más específicamente, el sistema descrito potencia la escalabilidad, fiabilidad, disponibilidad y tolerancia a fallos de una máquina de estado replicada y/o repositorio de información replicada en una arquitectura de sistema informático distribuido. Por consiguiente, el sistema descrito

supera ventajosamente una o más desventajas asociadas a enfoques convencionales para implementar una máquina de estado replicada y/o un repositorio de información replicada en una arquitectura de sistema informático distribuido.

Una máquina de estado replicada puede comprender un gestor de propuestas, un gestor de acuerdos y un temporizador de colisión/retroceso y un recuperador de almacenamiento. El gestor de propuestas facilita la gestión de propuestas emitidas por un nodo de una aplicación distribuida para posibilitar la ejecución coordinada de las propuestas por todos los nodos de la aplicación distribuida que necesitan hacer eso, posiblemente, pero no necesariamente, incluyéndose a sí mismo. El gestor de acuerdos facilita el acuerdo sobre las propuestas. El temporizador de colisión/retroceso impide anticipaciones repetidas de rondas al intentar conseguir acuerdo sobre las propuestas. El recuperador de almacenamiento recupera almacenamiento persistente utilizado para almacenar acuerdos de propuesta y/o las propuestas.

Una arquitectura de sistema informático distribuido puede comprender un sistema de red y una pluralidad de sistemas informáticos distribuidos interconectados mediante el sistema de red. Cada uno de los sistemas informáticos distribuidos puede incluir una respectiva máquina de estado replicada y un respectivo nodo de aplicación local conectado a la respectiva máquina de estado replicada. La respectiva máquina de estado replicada de cada uno de los sistemas informáticos distribuidos facilita la gestión de propuestas para posibilitar la ejecución coordinada de las propuestas por el nodo de aplicación distribuido de todos los sistemas informáticos distribuidos, facilita el acuerdo sobre las propuestas, impide anticipaciones repetidas de rondas al intentar conseguir acuerdo sobre las propuestas y recupera almacenamiento persistente utilizado para almacenar al menos uno de los acuerdos de propuesta y las propuestas.

Un método puede comprender una pluralidad de operaciones. Una operación puede realizarse para facilitar el acuerdo sobre propuestas recibidas de un nodo de aplicación local. Una operación puede realizarse para impedir anticipaciones repetidas de rondas al intentar conseguir acuerdo sobre las propuestas. Una operación puede realizarse para recuperar respectivo almacenamiento persistente utilizado para almacenar al menos uno de los acuerdos de propuesta y las propuestas.

Al menos una porción de las propuestas incluye etapas propuestas que corresponden a la implementación de una actualización de información iniciada por un nodo de una aplicación distribuida. Un orden de emisión de las propuestas puede conservarse mientras se facilita el acuerdo concurrente sobre las propuestas. Una porción de las propuestas puede proponer etapas de escritura que corresponden a una respectiva actualización de información y el gestor de propuestas puede asignar un número de secuencia local a cada una de las etapas de escritura propuestas y crear una intercalación globalmente única de las etapas de escritura propuestas de manera que todos los nodos de una aplicación distribuida que ejecutan las etapas de escritura propuestas ejecutan las etapas de escritura propuestas en una secuencia común. Puede proporcionarse un secuenciador local que incluye una pluralidad de entradas, cada una asociada a una respectiva de las propuestas, como puede ser un secuenciador global que incluye una pluralidad de entradas, referenciando cada una una respectiva de las entradas en el secuenciador local. Cada una de las entradas del secuenciador local puede tener un número de secuencia local único asignado a la misma, cada una de las entradas del secuenciador local puede estar dispuesta secuencialmente con respecto al número de secuencia local asignado y, después de que el gestor de acuerdos facilite el acuerdo sobre una de las propuestas, puede crearse una entrada que corresponde a la propuesta en la que se facilita el acuerdo dentro el secuenciador global en respuesta a la determinación de una posición en la que está situada la entrada dentro del secuenciador global. El recuperador de almacenamiento puede recuperar almacenamiento persistente borrando un registro para la propuesta de almacenamiento de propuesta persistente después de que se determine la posición de la entrada en el secuenciador global y sea conocida para todos los nodos. El temporizador de colisión/retroceso puede estar configurado para impedir anticipaciones repetidas realizando una operación de espera durante una duración de retardo de anticipación calculada para pasar después de iniciar una actual de las rondas para un primer proponente antes de iniciar una siguiente de la ronda para el primer proponente y/o una operación de espera durante un retardo de ronda en progreso calculado para que pase después de iniciar una actual de las rondas para el primer proponente antes de iniciar una siguiente de las rondas para un segundo proponente.

Volviendo ahora a las figuras, la Figura 1 muestra una arquitectura de sistema informático de múltiples sitios (es decir, denominado en el presente documento como la arquitectura de sistema informático de múltiples sitios 100) que puede incluir una pluralidad de sistemas de aplicación distribuidos 105 interconectados por una red de área extensa (WAN) 110. Cada uno de los sistemas de aplicación distribuidos 105 puede incluir una pluralidad de nodos de aplicación distribuidos 115 (por ejemplo, una aplicación que se ejecuta en una estación de trabajo), un replicador 120 y una réplica de repositorio 125. El replicador 120 de cada sistema de aplicación distribuido 105 puede conectarse entre la WAN 110, los nodos de aplicación distribuidos 115 del respectivo sistema de aplicación distribuido 105 y la réplica de repositorio 125 del respectivo sistema de aplicación distribuido 105.

En una disposición, cada réplica de repositorio 125 es un repositorio de Sistema de Versiones Concurrentes (CVS). CVS es un sistema de generación de versiones de código fuente abierto conocido. CVS, como la mayoría de otros sistemas de generación de versiones de código fuente, está diseñado para ejecutarse como un servidor central en el que múltiples clientes CVS (por ejemplo, unos nodos de aplicación distribuidos 115) se conectan usando un

protocolo de CVS sobre, por ejemplo, el Protocolo de Control de Transmisión (TCP). El servidor de CVS, como se implementa, bifurca un proceso por conexión de cliente para manejar una solicitud de CVS de cada cliente. Por consiguiente, el replicador 120 y la réplica de repositorio 125 permiten múltiples réplicas de un repositorio de CVS. Aunque un repositorio de información de CVS es un ejemplo de un repositorio de información, la materia objeto de la presente divulgación es útil al replicar otros tipos de repositorios de información. Bases de datos y sistemas de ficheros son ejemplos de otros tipos de este tipo de repositorios de información. Por consiguiente, la utilidad y aplicabilidad no están limitadas a un tipo particular de repositorio de información.

Como se analiza a continuación en mayor detalle, cada replicador 120 puede estar configurado para escribir actualizaciones de información desde su respectivo sistema de aplicación distribuido 105 a la réplica de repositorio 125 de cada otro sistema de aplicación distribuido 105. Cada replicador 120 puede ser el intermediario que actúa como una pasarela de aplicación entre clientes CVS (es decir, un respectivo nodo de aplicación distribuido 115) y un servidor de CVS dado (es decir, la respectiva réplica de repositorio 125). Cada replicador 120 coordina con otros replicadores de pares para asegurar que todas las réplicas de repositorio 125 permanecen en sincronía entre sí.

A diferencia de las soluciones convencionales, la arquitectura de sistema informático de múltiples sitios 100 no se basa en un coordinador de transacción central que se sabe que es un punto de fallo único. La arquitectura de sistema informático de múltiples sitios 100 proporciona un enfoque único a la replicación activa-activa en tiempo real, que opera bajo el principio de equivalencia de una copia a través de todas las réplicas de repositorio de CVS de un sistema de aplicación distribuido. Por consiguiente, en una disposición, cada réplica de repositorio está en sincronía con cada otra réplica de repositorio en una manera en tiempo real, por lo que los usuarios en cada nodo del sistema de aplicación distribuido (es decir, nodo de aplicación distribuido) están siempre trabajando desde la misma base de información (por ejemplo, programadores que trabajan desde la misma base de código).

A través de la integración del replicador 120 con la respectiva réplica de repositorio 125, cada réplica de repositorio se vuelve un nodo activo en la WAN 110 con su propio coordinador de transacción (es decir, el respectivo replicador 120). Cada coordinador de transacción distribuido acepta actualizaciones locales y las propaga a todas las otras réplicas de repositorio 125 en tiempo real. Por consiguiente, todos los usuarios dentro de la arquitectura de sistema informático de múltiples sitios 100 están trabajando de manera eficaz desde la misma información de repositorio (por ejemplo, un único repositorio de información de CVS) independientemente de la ubicación. Para este fin, una arquitectura de sistema informático de múltiples sitios como se describe en el presente documento es una solución de gestión de configuración de software (SCM) rentable, tolerante a fallos, que sincroniza trabajo de equipos de desarrollo distribuidos globalmente en tiempo real.

Cuando tengan lugar fallos de red o de servidor, los desarrolladores pueden continuar trabajando. Los cambios se registran en un registro diario de transacción del local de los replicadores 120. El registro diario de transacción es similar en función a un registro de rehacer de base de datos. Cuando se restaura la conectividad, el local de los replicadores 120 alcanza el replicador 120 de otros de los sistemas de aplicación distribuidos 105 para actualizar la local de las réplicas de repositorio 125, así como para aplicar los cambios capturados en el registro diario de transacción local mientras la red o el sistema estaban inactivos. La recuperación puede implementarse automáticamente, sin intervención alguna de un administrador de CVS. Esta capacidad de auto-restablecimiento asegura cero pérdida de datos, ningún tiempo de desarrollo perdido, y elimina el riesgo de error humano en un escenario de recuperación frente a desastres.

Los beneficios de trabajar desde esencialmente la misma información de repositorio incluyen no tener que cambiar procedimientos de desarrollo cuando el desarrollo se muda al extranjero, no tener que permanecer en espera mientras se espera que se completen las grandes construcciones cuando se está integrando trabajo de múltiples sitios, pudiendo detectar problemas de desarrollo antes y gastando menos recursos (por ejemplo, reduciendo la utilización de recursos redundante) en la aseguración de la calidad. Además, la recuperación de desastres no es un problema puesto que la capacidad de auto-restablecimiento integrada proporciona evitación frente a desastres. El trabajo nunca se pierde cuando un sistema está inactivo.

Como se ha desvelado anteriormente, la implementación de una máquina de estado replicada impacta ventajosamente a la escalabilidad, fiabilidad, disponibilidad y tolerancia a fallos de una máquina de estado replicada de este tipo. Impactando ventajosamente la escalabilidad, fiabilidad, disponibilidad y tolerancia a fallos, la disposición proporciona un enfoque práctico para implementar una máquina de estado replicada en una arquitectura de sistema informático de múltiples sitios. Al implementar una máquina de estado replicada como se describe en el presente documento, se cumplirán todos o una porción de los siguientes objetos: permitir que los nodos de un sistema informático distribuido de ordenadores evolucionen su estado en una manera coordinada; permitir que se conserve la consistencia de un sistema distribuido de ordenadores a pesar de fallos arbitrarios o fallos parciales de las redes informáticas, ordenadores o recursos informáticos; permitir que se cree un sistema fiable de nodos de aplicación distribuidos de componentes con fiabilidad modesta; asegurar la terminación del protocolo de acuerdo con la probabilidad como una función de tiempo que se acerca asintóticamente a 1, a pesar de las colisiones en el protocolo de acuerdo; eliminar colisiones en el protocolo de acuerdo bajo condiciones de operación normales; mejorar la eficacia del protocolo de acuerdo; reducir y limitar el uso de memoria y disco de la máquina de estado replicada; reducir el uso de recursos de red por la máquina de estado replicada; aumentar el rendimiento de

transiciones de estado realizable por la máquina de estado replicada; y posibilitar una gestión más eficaz de recursos de memoria y disco por los nodos de aplicación distribuidos servidos por la máquina de estado replicada.

Como se muestra en la Figura 2, se facilita la funcionalidad informática de múltiples sitios por una pluralidad de máquinas de estado replicadas 200 que interactúan entre sí y con un respectivo nodo de aplicación local 205 a través de un sistema de red 210. Preferentemente, pero no necesariamente, cada nodo de aplicación local 205 puede ser el de una aplicación distribuida y sirve como un proponente de propuesta o aceptador de propuesta en cualquier punto en el tiempo dado. En una realización, el sistema de red 210 puede incluir una Red de Área Extensa (WAN) conectada entre las máquinas de estado replicadas 200 y una respectiva Red de Área Local (LAN) conectada entre cada máquina de estado replicada 200 y el respectivo nodo de aplicación local 205. Por ejemplo, cada máquina de estado replicada 200 y su respectivo nodo de aplicación local 205 están situados en un respectivo sitio para un proyecto informático colaborativo de múltiples sitios. La porción de LAN del sistema de red 210 facilita la compartición de información en una base local (es decir, entre cada máquina de estado replicada 200 y su respectivo nodo de aplicación local 205) y la porción de WAN del sistema de 210 facilita la compartición de información en una base global (es decir, entre las máquinas de estado replicadas 200). Aunque una LAN, una WAN o ambas son ejemplos de componentes constituyentes de un sistema de red, las realizaciones no están limitadas a una configuración particular del sistema de red. Por ejemplo, otros sistemas de red incluyen un sistema de red ad-hoc que incluye ordenadores embebidos en un automóvil, un sistema de red que comprende una pluralidad de subredes en un centro de datos y un sistema de red que incluye una subred dentro de un centro de datos.

La Figura 3 es un diagrama de bloques que muestra componentes funcionales de cada máquina de estado replicada 200 mostrada en la Figura 2. Cada máquina de estado replicada 200 puede incluir un gestor de propuestas 220, almacenamiento de propuesta de persistencia 230, un gestor de acuerdos 240, un almacén de acuerdos 245, una capa de Protocolo de Transferencia de Ficheros Distribuido (DFTP) 250, un temporizador de colisión y repliegue 260, un secuenciador local 270, un secuenciador global 280 y un recuperador de almacenamiento 290 (es decir, un recolector de basura de almacenamiento persistente). El gestor de propuestas 220, el almacenamiento de propuesta de persistencia 230, el gestor de acuerdos 240, el almacén de acuerdos 245, la capa de DFTP 250, el temporizador de colisión y repliegue 260, el secuenciador local 270, el secuenciador global 280 y el recuperador de almacenamiento 290 están interconectados en al menos una porción entre sí para posibilitar la interacción entre ellos. Como se observará en el siguiente análisis, cada uno de los componentes funcionales de la máquina de estado replicada soporta ventajosamente funcionalidad.

Gestión de propuesta

Cada nodo de aplicación local 205 propone una secuencia de propuestas a la respectiva máquina de estado replicada 200. Las secuencia de propuestas propuesta por cada nodo local 6 constituye una secuencia local de ese respectivo nodo local 205, que puede mantenerse dentro del secuenciador local 270 de la respectiva máquina de estado replicada 200. El gestor de propuestas 220 de cada máquina de estado replicada 200 organiza la respectiva secuencia de propuestas en una única respectiva secuencia de propuestas global, que puede mantenerse dentro del secuenciador global 280 de la respectiva máquina de estado replicada 200. Cada secuencia global de propuestas tiene las siguientes propiedades: cada propuesta de cada secuencia local tiene lugar exactamente una vez en la respectiva secuencia global, la ordenación relativa de cualesquiera dos propuestas en una secuencia local puede conservarse opcionalmente en la respectiva secuencia global, y las secuencias globales (con o sin ordenación local conservada) asociadas a todos los nodos de aplicación local 205 son idénticas.

Cuando un hilo del nodo de aplicación local 205 propone una propuesta (por ejemplo, etapas de escritura) a la respectiva máquina de estado replicada 200, la máquina de estado replicada 200 asigna un número de secuencia local a la propuesta. Esa máquina de estado replicada 200 a continuación determina un número de acuerdo para esa propuesta. Como se hará evidente a partir de los análisis a continuación, el número de acuerdo determina la posición de una respectiva propuesta en la secuencia global. La máquina de estado replicada 200 a continuación graba un registro de la propuesta en su almacenamiento de propuesta persistente 230. La máquina de estado replicada 200 a continuación devuelve el control del hilo del nodo de aplicación local de vuelta al nodo de aplicación local, por lo que el hilo puede estar disponible para su uso por la aplicación local, y no en espera mientras se ejecuta el protocolo de acuerdo. La máquina de estado replicada a continuación inicia un protocolo de acuerdo para la propuesta mediante el gestor de acuerdos 240. Cuando se termina el protocolo de acuerdo, la máquina de estado replicada 200 compara el acuerdo alcanzado por el protocolo de acuerdo con el acuerdo propuesto contenido dentro de la propuesta. Si el acuerdo alcanzado por el gestor de acuerdos 240 puede ser el mismo que el de la propuesta, la máquina de estado replicada 200 concluye el procesamiento de la propuesta. De otra manera, la máquina de estado replicada 200 intenta repetitivamente el acuerdo sobre la propuesta usando un nuevo número de acuerdo hasta que el acuerdo alcanzado por el gestor de acuerdos pueda ser el mismo que el de la propuesta. Tras la conclusión de un acuerdo, cada nodo de aplicación local 205 pone en cola la propuesta ahora acordada en su secuencia global. Posteriormente, cada nodo de aplicación local 205 de la aplicación distribuida deja de poner en cola y ejecuta las propuestas contenidas dentro de la secuencia global.

La Figura 4 muestra una propuesta que se denomina en el presente documento como la propuesta 300. La propuesta 300 puede incluir un identificador de propuesta 320 (es decir, un identificador de un nodo de aplicación

local), un número de secuencia local (LSN) 330, un número de secuencia global (GSN) 340, un número de acuerdo 350 y contenido de propuesta 360. Preferentemente, pero no necesariamente, las propuestas emitidas por cada nodo de aplicación local 205 tienen la estructura de la propuesta 300.

La Figura 5 muestra una secuencia local, que se denomina en el presente documento como la secuencia local 400. La secuencia local 400 puede incluir los contenidos de cada una de las propuestas para el respectivo nodo de aplicación local 205. Más específicamente, tales contenidos incluyen el identificador del proponente, el número de secuencia local (LSN), el número de secuencia global (GSN), el número de acuerdo y el contenido de la propuesta. Preferentemente, pero no necesariamente, la secuencia local asociada a cada máquina de estado replicada 200 tiene la estructura de la secuencia local 400.

La Figura 6 muestra una secuencia global, que se denomina en el presente documento como la secuencia global 500. La secuencia global puede incluir el número de secuencia global para una serie de propuestas y un manejador de secuencia local. En una realización, el manejador de secuencia local puede ser un puntero a la respectiva secuencia local (es decir, como se representa, la secuencia local 400). El manejador de la secuencia local puede ser una clave a una tabla de secuencias locales. Preferentemente, pero no necesariamente, la secuencia global asociada a cada máquina de estado replicada 200 tiene la estructura de la secuencia global 500.

Acuerdos concurrentes

Las máquinas de estado replicadas 200 representadas en las Figuras 2 y 3, que son máquinas de estado replicadas consistentes con las enseñanzas en el presente documento, incorporan un mecanismo de acuerdo concurrente que permite que el acuerdo sobre múltiples propuestas de un proponente progrese de manera concurrente mientras, opcionalmente, conserva el orden en el que el proponente emitió las propuestas. En contraste, las máquinas de estado replicadas convencionales intentan un acuerdo sobre una propuesta después de alcanzar un acuerdo sobre una propuesta anterior. Esta metodología de máquina de estado replicada convencional asegura que una máquina de estado replicada convencional conserva el orden local de las propuestas. Por lo tanto, si un proponente propone en primer lugar la propuesta A y a continuación propone la propuesta B, la máquina de estado replicada convencional asegura que la propuesta A se ha acordado y después la propuesta B. Sin embargo, a diferencia de una máquina de estado replicada que implementa un mecanismo de repliegue como se describe en el presente documento, esta metodología convencional ralentiza la operación de la máquina de estado replicada convencional ya que el acuerdo sobre la propuesta B no puede iniciarse hasta que la propuesta A haya alcanzado un acuerdo.

En una configuración particular del sistema, cada objeto (es decir, una entrada) en la secuencia global puede numerarse secuencialmente. El número asociado a un objeto en la secuencia global identifica su posición con relación a los otros objetos en la secuencia global. Por ejemplo, un objeto con número 5 precede un objeto con número 6 y puede ser precedido por un objeto con número 4. Adicionalmente, cada objeto en la secuencia global contiene un manejador a una secuencia local, tal como el manejador de secuencia local 400 mostrado en la Figura 5. Si la aplicación no requiere la conservación del orden de emisión (es decir, el orden según se emite desde la fuente), cada objeto en la secuencia global contiene la propuesta misma. En este caso, la propuesta puede obtenerse directamente de la secuencia global en lugar de indirectamente mediante la secuencia local. El manejo a la secuencia local puede ser un puntero a la secuencia local, de lo contrario el manejador a la secuencia local puede ser una clave a una tabla de secuencias locales.

Haciendo referencia ahora a las Figuras 2 y 3, cada secuencia local contiene las propuestas de la máquina de estado replicada 200 propuestas por uno de los proponentes de la máquina de estado replicada 200. Cada nodo de aplicación local 205 de la máquina de estado replicada 200 mantiene una secuencia local para cada uno de los proponentes asociados a la máquina de estado replicada 200. Los objetos en la secuencia local se numeran secuencialmente. El número asociado a un objeto en la secuencia local identifica su posición con relación a los otros objetos en la secuencia local. Por ejemplo, el objeto con número 5 precede el objeto con número 6 y puede ser precedido por el objeto con número 4. Cada objeto en la secuencia local contiene una propuesta de la máquina de estado replicada 200.

En cada nodo de aplicación local 205 de la máquina de estado replicada 200, después de que se haya alcanzado el acuerdo sobre una propuesta, la propuesta puede añadirse a la secuencia global. La identidad del proponente (por ejemplo, ID de proponente 320 en la Figura 4) puede usarse como la clave para buscar una secuencia local de la tabla de secuencias locales. El número de secuencia local (LSN) de la propuesta determina la posición de la propuesta en la secuencia local. La propuesta puede a continuación insertarse en la posición determinada en la secuencia local. El número de acuerdo de la propuesta (por ejemplo, el número de acuerdo 350 en la Figura 4) determina la posición de la propuesta en la secuencia global. Puede insertarse un manejador a la secuencia local en la posición determinada en la secuencia global (es decir, basándose en el número de acuerdo). El GSN es un campo de contabilidad opcional para asociar a la propuesta para designar la posición real de la propuesta en la secuencia global cuando se consume como se describe en el párrafo a continuación.

En una disposición, un hilo especializado consume la secuencia global. El hilo espera hasta que se rellena la siguiente posición en la secuencia global. El hilo a continuación extrae la secuencia local almacenada en esa

posición de la secuencia global. El hilo a continuación espera hasta que se rellena la siguiente posición en la secuencia local. El hilo a continuación extrae la propuesta de la máquina de estado replicada 200 almacenada en esa posición de la secuencia local. Un experto en la materia apreciará que las propuestas no se extraerán necesariamente de acuerdo con la secuencia de números de acuerdo, sino que se extraerán en exactamente la misma secuencia en todos los nodos de aplicación. Esta secuencia de extracción puede registrarse por conveniencia de contabilidad en el campo de GSN, pero de otra manera no es esencial para la operación de la máquina de estado replicada 200. Por ejemplo, supóngase que un nodo de aplicación (A) emite sus primeras dos propuestas a la máquina de estado replicada (LSN 1 y LSN 2). Supóngase además que la máquina de estado replicada alcanzó un acuerdo sobre LSN 2 antes de alcanzar el acuerdo sobre LSN 1. Por lo tanto, el número de acuerdo para A:1 (LSN 1 del nodo de aplicación A) es 27 y el número de acuerdo para LSN 2 es 26 (es decir, hubo un total de 25 acuerdos anteriores sobre las propuestas de otros nodos de aplicación y ningún acuerdo intermedio sobre las propuestas de otros nodos de aplicación entre A:1 y A:2). Usando el método anterior, A:1 se extraerá de la secuencia global en la posición 26, y A:2 en la posición 27. Por lo tanto, el GSN respetará el orden LSN, pero el número de acuerdo no necesita necesariamente hacer esto. Esta metodología posibilita que una máquina de estado replicada como se describe en el presente documento procese los acuerdos de manera concurrente.

El hilo a continuación aplica la propuesta de la máquina de estado replicada 200. La aplicación de la propuesta puede conseguirse invocando una función de devolución de llamada registrada por una aplicación de la máquina de estado replicada 200.

Evitación de retroceso y colisión

Una máquina de estado replicada (por ejemplo, la máquina de estado replicada 200) puede incluir en consecuencia un mecanismo de retroceso para evitar la anticipación repetida de los proponentes (por ejemplo, nodos de aplicación local 205) en el protocolo de acuerdo del gestor de acuerdos 240. En contraste, cuando una ronda iniciada por un primer proponente se anticipa a una ronda iniciada por un segundo proponente, unas máquinas de estado replicadas convencionales permiten que el proponente anticipado inicie inmediatamente una nueva ronda con un número redondeado más alto que el del preferente. De manera indeseable, esta metodología convencional establece la etapa de anticipaciones repetidas de rondas, que puede conducir a que un protocolo de acuerdo se lleve a cabo durante un tiempo inaceptablemente largo (por ejemplo, de manera perpetua).

Al facilitar el retroceso, cuando se anticipa una ronda, el proponente calcula la duración de un retado de anticipación. El proponente a continuación espera esa duración calculada antes de iniciar la siguiente ronda de acuerdo con un algoritmo convencional para iniciar una ronda siguiente de este tipo.

Al facilitar la evitación de colisión, cuando un primer proponente detecta que un segundo proponente ha iniciado una ronda, el primer proponente calcula la duración de un retardo de ronda en progreso. El primer proponente se abstiene de iniciar una ronda hasta que se haya agotado la duración del retardo calculado.

En una disposición, un retardo dado crece exponencialmente con anticipaciones posteriores de una ronda. Además, el retardo preferentemente está aleatorizado.

Hay varios métodos posibles que pueden usarse para determinar la duración de un retardo dado. Una fuente de inspiración para métodos viables es la bibliografía sobre protocolos de Acceso Múltiple por Detección de Portadora/Detección de Colisión (CSMA/CD) para Ethernet sin conmutación. Un protocolo de CSMA/CD es un conjunto de reglas que determinan cómo responden los dispositivos de red cuando dos dispositivos de red intentan usar un canal de datos simultáneamente.

En una de varias posibles disposiciones, el siguiente método determina la duración de un retardo calculado. Un administrador que despliega la máquina de estado replicada 200 configura cuatro valores numéricos. Para el fin de la descripción de esta realización, los valores se denominan A, U, R y X. En una configuración válida, el valor R es mayor que cero, y menor que uno; el valor A es mayor que cero; el valor X es mayor que uno; el valor U es mayor que el valor A. El tiempo de ejecución del protocolo de acuerdo puede estimarse. Uno de varios posibles estimadores del tiempo de ejecución del protocolo de acuerdo puede ser una media de ventana móvil de tiempos de ejecución pasados del protocolo de acuerdo. Para el fin de este análisis, este valor estimado se denominará E. A se multiplica por U para determinar el valor M. Se selecciona el mayor de los dos valores A y E. Para el fin de este análisis, este valor seleccionado se denomina F. F se multiplica por X para determinar el valor C. Un valor aleatorio V se genera a partir de una distribución uniforme entre cero y C veces R. Si C es mayor que M, V se resta de C para calcular D. De otra manera, V se añade a C para calcular D.

El valor calculado D puede usarse como el retardo de la ronda en progreso. Puede usarse también como el retardo de anticipación la primera vez que se anticipa un nodo de aplicación local 205 en la ejecución de una instancia de protocolo de acuerdo. Cada vez posterior que el nodo de aplicación local 205 puede anticiparse en la ejecución de la instancia de protocolo de acuerdo, puede calcularse un nuevo valor D usando el valor antiguo D en lugar del valor A en el método anterior. El nuevo valor D puede usarse como el retardo de anticipación.

Recuperar almacenamiento persistente

Una máquina de estado replicada como se describe en el presente documento (por ejemplo, la máquina de estado replicada 200) recupera almacenamiento persistente usado para asegurar su tolerancia a fallos y alta disponibilidad. Haciendo referencia a las Figuras 2 y 3, el recuperador de almacenamiento 290 borra un registro de una propuesta propuesta del almacén de propuestas 230 después de que la máquina de estado replicada 200 ha determinado la posición de la propuesta propuesta en la secuencia global y todos los nodos de aplicación son informados de esta posición. A intervalos periódicos, cada nodo de aplicación local 205 envía un mensaje a cada uno de los otros nodos locales 205 que indica la posición rellenada de manera contigua más alta en su copia de la secuencia global. A intervalos periódicos, el recuperador de almacenamiento 290 borra todos los acuerdos hasta la posición rellenada de manera contigua más alta en todas las copias de la secuencia global que ya no se requieren por el nodo de aplicación local. De esta manera, cada máquina de estado replicada 200 recupera almacenamiento persistente.

Reservas débiles

Una máquina de estado replicada (por ejemplo, la máquina de estado replicada 200) proporciona un mecanismo de reserva débil opcional para eliminar la anticipación de proponentes bajo condiciones de operación normales. Haciendo referencia a las Figuras 2 y 3, cada proponente que controla una respectiva máquina de estado replicada 200 puede numerarse de manera contigua. Por ejemplo, si hay tres proponentes, pueden numerarse 1, 2, y 3. Un número de proponente determina qué propuestas de la respectiva máquina de estado replicada 200 controlará un correspondiente proponente. Si un número del proponente es M, y si hay N proponentes, el proponente controlará las propuestas numeradas M+(k.veces.N) (es decir, M más k multiplicado por N, para todos los valores enteros de k mayores o iguales a 0). Para permitir que un sistema de aplicación distribuido haga progreso cuando todos los proponentes de tal sistema no están disponibles, si una propuesta de la máquina de estado replicada 200 no puede determinarse en una manera oportuna, cualquier proponente asociado a la respectiva máquina de estado replicada 200 puede proponer "sin operación" (es decir, no-op) para esa propuesta. Para hacer esta optimización transparente a la aplicación distribuida, la máquina de estado replicada 200 no entrega las propuestas no-op a la aplicación distribuida. Sin operación hace referencia a una etapa de cálculo que, en general, no tiene ningún efecto, y en particular, no cambia el estado de la máquina de estado replicada asociada.

Números redondeados distinguidos y equitativos

El uso de la máquina de estado replicada como se describe en el presente documento asegura que uno de una pluralidad de proponentes competidores no se le dará anticipación cuando usa el mismo número redondeado para propuestas competidoras. En contraste, las máquinas de estado replicadas convencionales no incluyen un mecanismo que asegure que uno de una pluralidad de proponentes competidores no se anticipará cuando usa el mismo número redondeado para propuestas competidoras. Un número redondeado en tales máquinas de estado replicadas convencionales también puede ser un valor monotónico, que hace posible que se anticipe a todos los proponentes.

Además del componente monotónico, el número redondeado puede contener un componente distinguido. Un número entero distinto pequeño puede estar asociado a cada proponente de cada máquina de estado replicada 200. El número entero distinto sirve para resolver conflictos a favor del proponente con el componente distinguido más alto. Además del componente monotónico y el componente distinguido, el número redondeado contiene un componente aleatorio. Un número redondeado de esta forma asegura que uno de una pluralidad de proponentes competidores no se anticipará cuando se usa el mismo número redondeado para propuestas competidoras (es decir, mediante el componente distinto del número redondeado) y asegura que la resolución de conflictos no favorece o deja de favorecer perpetuamente uno particular de los proponentes (es decir, mediante el componente aleatorio del número redondeado).

Un mecanismo para comparar dos números redondeados opera como sigue. El número redondeado con el componente monotónico más grande es mayor que el otro. Si los componentes monotónicos de los dos números redondeados son iguales, el número redondeado con el componente aleatorio mayor es más grande que el otro. Si las dos comparaciones anteriores no distinguen los números redondeados, el número redondeado con el componente distinguido mayor es más grande que el otro. Si las tres comparaciones anteriores no distinguen los números redondeados, los números redondeados son iguales.

Recuperar almacenamiento persistente de manera eficaz

Haciendo referencia a las Figuras 3 y 4, los registros en el almacén de propuestas persistente 230 de una máquina de estado replicada 200 se organizan en grupos. Cada grupo almacena registros de propuestas propuestas con números de secuencia local contiguos 330. Por ejemplo, los registros con números de secuencia locales N. º 1 a N. º 10000 pueden pertenecer al grupo 1, los registros con números de secuencia local N. º 10001 a N. º 20000 pueden pertenecer al grupo 2, y así sucesivamente.

Haciendo referencia a grupos de propuestas persistentes, cada grupo puede almacenare de tal manera que los

recursos de almacenamiento usados por todo el grupo puedan recuperarse eficazmente. Por ejemplo, en un sistema de almacenamiento basado en ficheros, cada grupo usa su propio fichero o conjunto de ficheros.

Haciendo referencia aún a grupos de propuestas persistentes, el recuperador de almacenamiento 290 rastrea solicitudes para borrar registros individuales, pero no borra registros individuales en el momento de las solicitudes. Cuando las solicitudes acumuladas para borrar registros individuales incluyen todos los registros en un grupo, el recuperador de almacenamiento 290 recupera eficazmente los recursos de almacenamiento usados por el grupo. Por ejemplo, en un sistema de almacenamiento basado en ficheros, el fichero o conjunto de ficheros usado por el grupo puede borrarse.

Los registros en el almacén de acuerdos 245 de la máquina de estado replicada 200 se organizan en grupos. Cada grupo almacena registros de instancias de protocolo de acuerdo con números de instancia de acuerdo contiguos 150. Por ejemplo, los registros con números de instancia de acuerdo N. º 1 a N. º 10000 pueden pertenecer al grupo 1, los registros con números de instancia de acuerdo N. º 10001 a N. º 20000 pueden pertenecer al grupo 2, y así sucesivamente.

Haciendo referencia a grupos de instancias de protocolo de acuerdo, cada grupo puede almacenarse de tal manera que los recursos de almacenamiento usados por el grupo entero pueden recuperarse eficazmente. Por ejemplo, en un sistema de almacenamiento basado en ficheros, cada grupo usa su propio fichero o conjunto de ficheros.

Haciendo referencia aún a grupos de instancias del protocolo de acuerdo, el recuperador de almacenamiento 290 rastrea solicitudes para borrar registros individuales, pero no borra registros individuales en el momento de las solicitudes. Cuando las solicitudes acumuladas para borrar registros individuales incluyen todos los registros en un grupo, el recuperador de almacenamiento 290 recupera eficazmente los recursos de almacenamiento usados por el grupo. Por ejemplo, en un sistema de almacenamiento basado en ficheros, el fichero o conjunto de ficheros usado por el grupo puede borrarse.

Manejar eficazmente propuestas pequeñas

Haciendo referencia a las Figuras 3 y 4, una máquina de estado replicada de acuerdo con una realización (por ejemplo, la máquina de estado replicada 200) pone en lotes la transmisión de las propuestas propuestas a la máquina de estado replicada 200 desde uno de origen de los nodos de aplicación local 205 a unos receptores de los nodos de aplicación local 205. Una práctica de este tipo permite que una máquina de estado replicada, como se expone en el presente documento, utilice eficazmente un protocolo de comunicación basado en paquetes en una situación donde el tamaño de las propuestas de la máquina de estado replicada es pequeño con relación al tamaño de un paquete de datos en el protocolo de comunicación basado en paquetes subyacente usado por la máquina de estado replicada.

Un lote de propuestas de este tipo puede tratarse como una única propuesta por el protocolo de acuerdo. De esta manera, en cada nodo local 205, mientras una respectiva máquina de estado replicada 200 está determinando el número de acuerdo 350 de un primer lote de propuestas propuestas, las propuestas propuestas en el respectivo nodo de aplicación local 205 pueden acumulares en un segundo lote de propuestas. Cuando se determina el número de acuerdo 150 del primer lote, la máquina de estado replicada 200 inicia la determinación del número de instancia de acuerdo 350 del segundo lote, y las propuestas propuestas en ese nodo de aplicación local 205 se acumulan en un tercer lote--y así sucesivamente.

Manejar eficazmente grandes propuestas 110

Para reducir el ancho de banda de red para propuestas grandes, una máquina de estado replicada como se describe en el presente documento permite que se etiqueten propuestas mediante un id de propuesta corto (por ejemplo, un id globalmente único de 16 bytes) y/o las propuestas pueden codificarse en un formato denominado como propuesta basada en fichero. En contraste, las propuestas grandes presentan un problema para máquinas de estado replicadas convencionales en que tales propuestas grandes se envían esencialmente múltiples veces a través de una red, según se acciona por el protocolo de acuerdo de una máquina de estado replicada convencional. Tal transmisión múltiple puede no preferirse puesto que el tamaño de propuestas grandes puede ser de varios megabytes o incluso gigabytes.

Cuando se transmiten propuestas grandes, una alternativa es transmitir únicamente identificadores de propuestas cortos una vez que se ha transmitido satisfactoriamente la propuesta real a un punto terminal de red. Las propuestas basadas en fichero esencialmente llevan un puntero de fichero en memoria mientras que el contenido de propuesta real puede mantenerse en disco en un fichero. Cuando se transporta una propuesta basada en fichero de este tipo, una máquina de estado replicada de acuerdo con una realización usa un protocolo de envío por flujo continuo de fichero tolerante a fallos eficaz. Tal transporte puede manejarse por la capa de DFTP 250 de una máquina de estado replicada 200 (Figura 3). La capa de DFTP 250 rastrea la propuesta basada en ficheros de pares y un punto terminal de red. Asegura que una propuesta basada en fichero se transmita únicamente una vez a un punto terminal de red. En el caso de fallos que conduzcan a transferencias parciales, la propuesta basada en fichero puede recuperarse

desde cualquier punto terminal disponible que tenga la porción requerida del fichero.

La implementación de DFTP puede usar ficheros de envío de ficheros nativo o de memoria mapeada para la transferencia de ficheros eficaz si el sistema operativo soporta estas características. Si el emisor original no es alcanzable por un nodo que requiera un fichero, ese nodo ubicará un emisor alternativo--un nodo diferente en el sistema que pasa a tener el fichero. Cuando se opera sobre el protocolo de TCP, DFTP usa múltiples conexiones de TCP para aprovecharse mejor de las conexiones de ancho de banda alto que también son objeto de alta latencia. Además, para aprovecharse mejor de las conexiones de ancho de banda alto que son también objeto de alta latencia, puede ajustarse de manera apropiada y/o deseable el protocolo de TCP.

Volviendo ahora a un análisis de replicación escalable y activa de repositorios de información, la implementación de tal replicación puede utilizar la máquina de estado replicada anteriormente mencionada. Más específicamente, proporcionar tal replicación impacta ventajosamente a la escalabilidad, fiabilidad, disponibilidad y tolerancia a fallos de una máquina de estado replicada de este tipo. Por consiguiente, la implementación de una máquina de estado replicada impacta ventajosamente tal replicación en una arquitectura de sistema informático distribuido. Al implementar la replicación de un repositorio de información, se cumplirán todos o una porción de los siguientes objetos: posibilitar la replicación de un repositorio de CVS, una base de datos, o cualquier repositorio de información en general; permitir uso concurrente, incluyendo la modificación, de todas las réplicas de un repositorio de información; conservar la consistencia de las réplicas a pesar de fallos esencialmente arbitrarios o fallos parciales de las redes informáticas usadas en la infraestructura de replicación; conservar la consistencia de las réplicas a pesar de fallos esencialmente arbitrarios o fallos parciales de los ordenadores o recursos informáticos asociados a las réplicas; asegurar la disponibilidad continua del repositorio de información a pesar de fallos significativos de la naturaleza anteriormente descrita; permitir la distribución geográfica de réplicas de manera que no haya restricciones sobre cómo de lejos (por ejemplo, en diferentes continentes) o cómo de cerca (por ejemplo, en el mismo centro de datos, o incluso en el mismo bastidor) están las réplicas unas de las otras; permitir todas las réplicas del repositorio de información en conjunto para manejar una carga superior que la que puede manejarse por una instancia del repositorio; conservar equivalencia de una copia de las réplicas; posibilitar la replicación del repositorio de información sin introducir un único punto de fallo en el sistema; permitir la replicación de un repositorio de información sin modificaciones a las implementaciones del repositorio de información; permitir la replicación de un repositorio de información sin modificaciones a las implementaciones de los clientes del repositorio de información; ofrecer a los clientes de un repositorio de CVS tiempos de respuesta de un repositorio de CVS local co-ubicado mediante quórum rotativo de réplica; reducir la comunicación de red entre clientes de repositorio de CVS y repositorio de CVS remoto en un factor de aproximadamente 3 en una red de área extensa (por ejemplo, de aproximadamente 4,5 viajes de ida y vuelta a aproximadamente 1,5 viajes de ida y vuelta); permitir la recuperación remota de réplicas fallidas de una manera automatizada sin requerir intervención del administrador; y asegurar limpieza de estado distribuido de todas las réplicas de una manera automatizada sin requerir la intervención del administrador.

Haciendo referencia a la Figura 7, se muestra una realización de un replicador, que se denomina en el presente documento como el replicador 600. El replicador 600 consiste en una pluralidad de módulos funcionales, que incluyen una interfaz de cliente de replicador 610, un pre-calificador 620, una máquina de estado replicada 630, un planificador 640, una interfaz de repositorio de replicador 650, un manejador de resultado 660 y una consola de administrador 670. La interfaz de cliente de replicador 610, el pre-calificador 620, la máquina de estado replicada 630, el planificador 640, la interfaz de repositorio de replicador 650, el manejador de resultado 660 y la consola de administrador 670, cada uno están interconectados a al menos una porción de los otros módulos para posibilitar la interacción entre los mismos. La máquina de estado replicada 200, cuya funcionalidad se analizó con referencia a las Figuras 2-6, es un ejemplo de la máquina de estado replicada 630 del replicador 600. Por lo tanto, la máquina de estado replicada 630 es fiable, disponible, escalable y tolerante a fallos.

La Figura 8 muestra el despliegue del replicador 600 dentro de una arquitectura de sistema informático de múltiples sitios. La arquitectura de sistema informático de múltiples sitios puede incluir una pluralidad de sistemas de aplicación distribuidos 601. Cada sistema de aplicación distribuido 601 puede incluir una pluralidad de clientes 680, un replicador 600, una interfaz de cliente de repositorio 690, un repositorio 695 (es decir, un repositorio de información) y una red 699. La red 699, que generalmente no es necesariamente un componente de uno cualquiera de una pluralidad de sistemas de aplicación distribuidos 601, puede estar conectada entre los clientes 680 de cada sistema de aplicación distribuido 601 y el respectivo replicador 600 y entre la interfaz de cliente de repositorio 690 de cada sistema de aplicación distribuido 601 y el respectivo replicador 600, interconectando por lo tanto los clientes 680, el replicador 600 y el repositorio 695 de cada sistema de aplicación distribuido 601 para posibilitar la interacción de tales componentes de cada sistema de aplicación distribuido 601. La red puede estar también conectada entre el replicador 600 de todos los sistemas de aplicación distribuidos 601, posibilitando por lo tanto la interacción entre todos los sistemas de aplicación distribuidos 601. Las redes 699 pueden estar aisladas entre sí, pero no necesitan estarlo. Por ejemplo, la misma red puede satisfacer los tres papeles anteriormente desvelados.

Como se muestra en la Figura 8, tres clientes 680 están "cerca" de cada uno de los repositorios 695 (es decir, un elemento de sistema de los sistemas de aplicación distribuidos 601 que comprende un respectivo repositorio 695). Por cerca, se pretende que uno particular de los clientes 680 cerca de uno particular de los repositorios 695

preferiría acceder a ese uno particular de los repositorios 695. Como alternativa, ese uno particular de los clientes 680 podría acceder potencialmente al repositorio 695 de uno cualquiera de los sistemas de aplicación distribuidos 601.

Los operadores de un sistema informático distribuido incluyen los usuarios del cliente 680 y el administrador o administradores de los sistemas de aplicación distribuidos 601. Los usuarios del cliente 680 siguen las instrucciones de su manual de cliente de usuario. Un usuario podría permanecer ajeno al hecho de que está usando un replicador de acuerdo con una realización, ya que muchos de los aspectos ventajosos de las realizaciones pueden ser transparentes para el usuario. Un administrador, además de las tareas convencionales de administración del mismo repositorio 695, configurará las redes en consecuencia, según sea necesario y si fuera necesario para la operación.

Las máquinas de estado replicadas 630 de cada sistema de aplicación distribuido 601 se comunican entre sí a través de la red 699. Cada interfaz de repositorio de replicador 650 interactúa a través de la red 699 con el repositorio 695 del respectivo sistema de aplicación distribuido 601. El cliente 680 interactúa a través de la red 699 con la interfaz de cliente de replicador 610. Opcionalmente, puede usarse un producto, tal como, por ejemplo, Cisco Systems Director para posibilitar que un cliente particular 680 de uno particular de los sistemas de aplicación distribuidos 601 falle en cualquiera de los otros sistemas de aplicación distribuidos 601, si el sistema de aplicación distribuido 601 que comprende el cliente 680 puede no estar disponible en un momento particular para proporcionar una funcionalidad requerida.

Haciendo referencia ahora a las Figuras 7 y 8, la interfaz de cliente de replicador 610 puede ser responsable de hacer de interfaz con uno particular de los clientes 680 (es decir, el cliente particular 680) asociado a un repositorio dirigido 695. La interfaz de cliente de replicador 610 reconstruye los comandos emitidos por el cliente particular 680 a través de la red 699 y entrega los comandos al pre-calificador 620. El pre-calificador 620 posibilita la operación eficaz del replicador 600, pero puede no requerirse para la operación útil y ventajosa del replicador 600.

Para cada comando, el pre-calificador 620 puede determinar opcionalmente si el comando está condenado a fallar, y, en caso afirmativo, determinar un mensaje de error apropiado o estado de error a devolverse al cliente particular 680. En caso afirmativo, ese mensaje de error o estado de error puede devolverse a la interfaz de cliente de replicador 610 y la interfaz de cliente de replicador 610 entrega ese mensaje de error o estado de error al cliente particular 680. Posteriormente, el comando ya no puede procesarse más por el replicador 600.

Para cada comando, el pre-calificador 620 puede determinar opcionalmente si el comando puede omitir la máquina de estado replicada 630 o tanto la máquina de estado replicada 630 como el planificador 640. Si el pre-calificador 620 no determina que la máquina de estado replicada 630 pudiera omitirse, el comando puede entregarse a la máquina de estado replicada 630. La máquina de estado replicada 630 recopila todos los comandos emitidos para ella y sus máquinas de estado replicadas de pares 630 en cada otro replicador asociado 600 del sistema de aplicación distribuido 601. Esta secuencia de operaciones puede asegurarse que es idéntica en todos los sistemas de aplicación distribuidos 601. En cada uno de los sistemas de aplicación distribuidos 601, la respectiva máquina de estado replicada 630 entrega los comandos recopilados como anteriormente, en secuencia, al respectivo planificador 640.

El planificador 640 realiza un análisis de dependencia en los comandos entregados a él, y determina la ordenación parcial más débil de comandos que aseguraría aún capacidad de serialización de una copia. Tal análisis de dependencia y capacidad de serialización de una copia se desvela en la referencia de la técnica anterior de Wesley Addison titulada "Concurrent Control & Recovery in Database Systems" y publicada en un libro de referencia por P. Berstein et. al. El planificador 640 a continuación entrega los comandos a la interfaz de repositorio de replicador 650, concurrentemente cuando se permite por el orden parcial construido, de lo contrario de manera secuencial.

La interfaz de repositorio de replicador 650 entrega los comandos al repositorio 695. En respuesta, se produce uno de tres resultados. Posteriormente, la interfaz de repositorio de replicador 650 entrega el resultado producido al manejador de resultado 660.

Uno primero de los resultados puede incluir que el repositorio 695 devuelva una respuesta al comando. Esta respuesta contiene un resultado, un estado, o ambos, que indica que nada fue incorrecto durante la ejecución del comando. Si el comando se originó localmente, el manejador de resultado 660 entrega la respuesta a la interfaz de cliente de replicador 610, que a su vez entrega la respuesta al cliente 680. Si el comando se originó en un replicador de un sistema de aplicación distribuido diferente 601, la respuesta preferentemente se descarta.

Uno segundo de los resultados puede incluir que el repositorio 695 responda con un estado de error. El manejador de resultado 660 determina si el estado de error indica un error determinístico en el repositorio 695 (es decir, si tuviera lugar el mismo error o uno comparable en cada uno de los otros sistemas de aplicación distribuidos 601). Si la determinación del error puede ser ambigua, el manejador de resultado 660 intenta comparar el error con el resultado en otros sistemas de aplicación distribuidos 601. Si esto no resuelve la ambigüedad, o si el error puede ser inequívocamente no determinista, el manejador de resultado 660 suspenderá la operación del replicador 600 e informará al operador mediante la consola de administrador 670 (es decir, mediante la emisión de una notificación

mediante la consola administrativa 670).

En el caso donde el replicador sea un replicador de CVS, como se analiza a continuación con referencia a funcionalidad específica de CVS, puede usarse una lista de patrones de errores por el manejador de resultado para etiquetar el error determinístico. El manejador de resultado 660 usa estos patrones para hacer una coincidencia de expresión regular en el flujo de respuesta.

Uno tercero de los resultados puede incluir que se cuelgue el repositorio 695 (es decir, que no vuelva de la ejecución del comando). Este resultado puede tratarse exactamente como un error no determinístico como se analiza en referencia al segundo de los resultados.

Cada replicador 600 puede configurarse de manera alternativa. El replicador 600 puede estar embebido en y accionarse directamente por el cliente 680 del repositorio 695. En otra alternativa, el replicador 600 puede estar embebido en la interfaz de cliente 690 al repositorio 695. En otra alternativa, el replicador 600 puede estar embebido en el repositorio 695. En otra alternativa, el secuenciador global del replicador (por ejemplo, el secuenciador global 280 mostrado en la máquina de estado replicada 200 en la Figura 3) puede estar basado en otras tecnologías, con compromisos correspondientes de robustez y calidad de servicio. Uno de varios ejemplos posibles de una tecnología de este tipo es la comunicación de grupo. En otra alternativa, el replicador 600 acciona más de un repositorio 695, con correspondiente compromiso de robustez y calidad de servicio. En otra alternativa, los módulos del replicador 600 se unen en módulos con resolución más basta, se dividen en módulos con resolución más precisa, o ambas. En otra alternativa, como una salvaguarda redundante contra la desviación de la capacidad de serialización de una copia, se comparan las respuestas de todos los sistemas de aplicación distribuidos 601 para asegurar que la información contenida en los repositorios 695 de cada sistema de aplicación distribuido 601 permanece consistente con respecto a cada otro sistema de aplicación distribuido 601.

En referencia a las Figuras 7 y 8, cada uno de los repositorios 695 analizados anteriormente puede ser un repositorio de Sistema de Versiones Concurrentes (CVS) y los clientes 680 pueden ser, en correspondencia, clientes de CVS. Cuando los repositorios 695 son repositorios de CVS y los clientes 680 son clientes de CVS, las interfaces asociadas a los repositorios 695 y los clientes 680 son interfaces específicas de CVS (por ejemplo, una interfaz de cliente de CVS de replicador, una interfaz de repositorio de CVS de replicador y una interfaz de cliente de CVS de repositorio). Adicionalmente, el replicador 600 puede modificarse para incluir funcionalidad que está específicamente y especialmente configurada para su uso con un repositorio de CVS.

La interfaz de cliente de replicador 610 desvelada en el presente documento puede estar configurada específicamente para hacer de interfaz con un cliente de CVS de un repositorio de CVS dirigido. Para este fin, la interfaz de cliente de replicador 610 almacena bytes entrantes del cliente de CVS en una memoria intermedia de fichero de memoria mapeada. La interfaz de cliente de replicador 610 detecta el final del comando de CVS cuando observa una cadena de comando válida en el flujo de bytes entrante. Una lista, no limitante, de tales cadenas de comandos válidas puede incluir, pero sin limitación, "Root", "respuestas válidas", "solicitudes válidas", "Repository", "Directory", "Max-dotdot", "Static-directory", "Sticky", "Entry", "Kopt", "Checkin-time", "Modified", "Is-modified", "UseUnchanged", "Unchanged", "Notify", "Questionable", "Argument", "Argumentx", "Global_option", "Gzip-stream", "wrapper-sendme-rcsOptions", "Set", "expand-modules", "ci", "co", "update", "diff', "log", "rlog", "list", "rlist", "global-list-quiet", "Is", "add", "remove", "update-patches", "gzip-file-contents", "status", "rdiff", "tag", "rtag", "import", "admin", "export", "history", "release", "watch-on", "watch-off, "watch-add", "watch-remove", "watchers", "editors", "init", "annotate", "ranno tate", "noop" y "version".

La interfaz de cliente de replicador 610 a continuación intenta clasificar el comando de CVS entrante como un comando de lectura o un comando de escritura. Una lista, no limitante, de cadenas de comando de escritura válidas puede incluir, pero sin limitación, "ci", "tag", "rtag", "admin", "import", "add", "remove", "watch-on", "watch-off' e "init". Cualquier comando dentro de la lista de cadenas de comandos válidas que no pertenece a la lista de cadenas de comandos de escritura válidas se considera en el presente documento que es una cadena de comandos de lectura con respecto a la lista de cadenas de comandos válidas.

Los comandos de lectura se entregan directamente a la interfaz de repositorio de replicador de CVS para su ejecución por el repositorio de CVS dirigido. Los comandos de escritura de CVS se entregan opcionalmente al módulo pre-calificador 20.

Para cada comando de escritura de CVS, el módulo pre-calificador 20 puede determinar opcionalmente si el comando de CVS está condenado a fallar, y, en caso afirmativo, determinar un mensaje de error apropiado o estado de error a devolverse al cliente de CVS. La detección de fallo puede estar basada en adaptar el resultado o flujo de bytes de estado devuelto por el repositorio de CVS con patrones de error conocidos. Ejemplos de patrones de error de sistema conocidos incluyen, pero sin limitación, no se puede crear enlace simbólico de .* para .*; no se puede iniciar servidor mediante rsh; no se puede realizar fstat .*; fallo al crear fichero temporal; no se puede abrir fichero dbm .* para creación; no se puede escribir en .*; no se puede registrar fichero histórico; no se puede abrir fichero histórico: .*; no se puede abrir '.*'; no se puede registrar fichero RCS .* para mapear; no se puede abrir fichero .* para comparar; memoria virtual agotada; no se puede realizar ftello en fichero RCS .*; no se puede leer .*; no se

puede obtener lista de grupos auxiliares; no se puede realizar fsync en fichero .* después de copiar; no se puede registrar .*; no se puede abrir directorio actual; no se puede registrar directorio .*; no se puede escribir .*; no se puede leer enlace.*; no se puede cerrar tubería; no se puede cambiar a directorio .*; no se puede crear fichero temporal; no se puede obtener información de fichero para .*; no se puede abrir fichero de salida diff .*; no se puede crear .*; no se puede obtener directorio de trabajo; no se puede realizar lstat .*; bifurcación para diff falló .*; no se puede obtener información para '.*'; no se puede cambiar modo para .*; no se puede realizar ftello para .*; Verificación de mensaje fallida; no se puede registrar fichero temporal .*; sin memoria; no se puede hacer directorio .* in .*; inicio de sesión: fallo al leer contraseña; error leyendo fichero histórico; no se puede obtener directorio de trabajo; no puede establecerse bandera cierre en ejecución activada \d+; error al escribir fichero de bloqueo .*; no se puede escribir en fichero histórico: .*; no se puede renombrar fichero .* to .*; no se puede cambiar a .* directorio; no se puede obtener información de fichero para .*; no se puede crear .* para copiar; no se puede escribir fichero temporal .*; no se puede abrir .*; lectura de control de flujo fallida; escribir a servidor; no se puede cerrar .*; no se puede abrir fichero de bloqueo '.*' no se puede realizar fdopen \d+ para lectura; no se puede cerrar fichero temporal .*; no se puede cambiar directorio a directorio de proceso de pago solicitado '.*'; no se puede hacer directorio.*; valor de umask inválido in; fallo a abrir .* para lectura; no se puede obtener número de grupos auxiliares; no se puede abrir .* para escritura; no se puede realizar chdir a .*; bifurcación fallida mientras se realiza diffing .*; no se puede abrir .*; no se puede realizar fdopen \d+ para escritura; escritura en .* fallida; no se puede crear fichero temporal .*; no se puede leer .*; no se puede escribir fichero .* para copiar; no se puede abrir .* para copiar; no se puede realizar dup2 tubería; no se puede realizar get-wd en .*; no se puede abrir .* para escritura; no se puede bifurcar; error escribiendo en servidor; no se puede comprobar en .* --bifurcación fallida; no se puede leer fichero.* para comparar; no se puede enlazar.* to.*; error al cerrar .*; no se puede realizar dup net conexión; lectura de datos fallida; no se puede leer .*; no se puede eliminar .*; no se puede realizar chdir a '.*'; no se puede abrir fichero temporal .*; no se puede registrar .*; no se puede abrir directorio .*; fwrite fallido; no se puede crear fichero temporal '.*'; no se puede registrar fichero temporal; no se puede registrar .*; no se puede leer '.*'; error al realizar diffing .*; no se puede crear fichero especial .*; no se puede cerrar fichero histórico: .*; no se puede mapear memoria a archivo de RCS *; no se puede hacer directorio '.*'; no se puede leer fichero .* para copiar; no se puede crear tubería; no se puede abrir fichero temporal .*; no se puede eliminar fichero .*; no se puede abrir; no se puede buscar fin de fichero histórico: .*; no se puede realizar chdir a .*; lectura de longitud fallida; no se puede ejecutar .*; no se puede realizar fdopen .* y no se puede hallar tamaño de fichero temporal. Ejemplos de patrones de error no de sistema conocidos incluyen, pero sin limitación, error interno; no tal repositorio; no se puede encontrar versión deseada; getsockname fallido:; advertencia: ferror establecido mientras se rescribía fichero RCS; error interno: islink no es como readlink; acceso denegado; no se puede comparar ficheros de dispositivo en este sistema; error interno de servidor: caso sin manejar en server_updated; recibida .* señal; error interno: sin información de revisión para; error de protocolo: duplicar modo; error interno de servidor: sin modo en servidor actualizado; rcsbuf caché abierta: error interno; Error fatal, abortar; error fatal: salir; .*: EOF no esperado; .*: número de revisión confundido; fichero rcs inválido; EOF en clave en fichero RCS; ficheros RCS en CVS siempre finalizan in,v; información de enlace definitivo perdida para; no se puede leer .*: fin de fichero; rcsbuf abierto: error interno; sin memoria; no se puede asignar infopath; bloqueos finales de .* inesperado; error interno: fecha incorrecta .*; autenticación kerberos fallida: .*;.*, delta .*: EOF no esperado; EOF no esperado leyendo fichero RCS .*; ERROR: sin espacio-abortar; EOF de control de flujo; no se puede realizar fseeko fichero RCS .*; fallo de suma de comprobación en .*; error interno de CVS: estado desconocido \d+; error interno: argumento incorrecto para run_print; no se puede copiar ficheros de dispositivo en este sistema; fin de fichero no esperado leyendo .*; sin memoria; error interno: no fichero RCS analizado; error interno: EOF demasiado pronto en RCS_copydeltas; error interno: soporte de prueba para respuesta desconocida\?; EOF en valor en fichero RCS .*; PANIC* ficheros de administración faltantes\!; fin de fichero prematuro leyendo .*; EOF mientras se buscaba valor en fichero RCS .*; no se puede continuar; bloqueo de lectura fallido-abandonar; lectura de EOF no esperada .*; no se puede resucitar '.*'; fichero RCS eliminado por segunda parte; su nombre de usuario aparente .* es desconocido para este sistema; corrupción de base de datos de atributo de fichero: pestaña perdida en .*; no se puede importar .*: no se puede importar ficheros de dispositivo en este sistema; no se puede importar .*: clase desconocida de fichero especial; no se puede importar .*: fichero especial de tipo desconocido; ERROR: no se puede realizar mkdir .* --no añadido; no se puede crear bloqueo de escritura en repositorio '.*; no se puede crear .*: no se puede crear ficheros especiales en este sistema; no se puede conservar .*: no se puede grabar ficheros de dispositivo en este sistema; error analizando fichero de repositorio .* fichero puede estar corrupto y estado de fichero desconocido \d+ para fichero .*.

Como se ha analizado anteriormente en referencia a las Figuras 7 y 8, para cada comando, el módulo pre-calificador 620 puede determinar que el comando está condenado a fallar y puede omitir tanto la máquina de estado replicada 630 como el planificador 640. En el caso de la funcionalidad específica de CVS, si el módulo pre-calificador 620 no determinó que la máquina de estado replicada 630 pudiera omitirse, el comando puede convertirse en un comando de propuesta de CVS. El comando de propuesta de CVS contiene el conjunto de bytes de comando de CVS real así como un conjunto de bloqueos que describe los bloqueos de escritura que provocaría este comando de CVS al repositorio de CVS para obtener si se ejecutara por él directamente. Como se analiza a continuación, el planificador 640 utiliza este conjunto de bloqueos.

El comando de propuesta de CVS puede entregarse a la máquina de estado replicada 630. La máquina de estado replicada 630 recopila todos los comandos emitidos para ella y sus máquinas de estado replicadas de pares 630 en cada uno de los otros replicadores, en una secuencia. Esta secuencia se asegura que es idéntica en todas las

réplicas. En cada uno de los sistemas de aplicación distribuidos 601, la máquina de estado replicada 630 entrega los comandos recopilados como anteriormente, en secuencia, al planificador 640.

El planificador 640 realiza un análisis de dependencia en los comandos entregados a él, y determina la ordenación parcial más débil de comandos que aseguraría aún capacidad de serialización de una copia. El planificador 640 entrega los comandos a la interfaz de repositorio de replicador de CVS, concurrentemente cuando se permite por el orden parcial construido, de lo contrario de manera secuencial.

El análisis de dependencia puede estar basado en ensayar conflictos de bloqueo. Cada comando de propuesta de CVS emitido al planificador contiene un conjunto de bloqueos. El planificador asegura que se entrega un comando a la interfaz de repositorio de CVS si, y únicamente si, ningún otro conjunto de bloqueos del comando entra en conflicto con su conjunto de bloqueos. Si se detecta un conflicto, el comando espera en cola para planificarse en un punto más tarde cuando todos los bloqueos en el conjunto de bloqueos puedan obtenerse sin conflictos.

Afiliaciones

En un sistema distribuido de máquinas de estado replicadas que alojan procesos, es deseable tener la capacidad de cambiar la asociación de la máquina de estado a la colección de procesos, o afiliación, que participa en la operación de la máquina de estado. Por ejemplo, un ejemplo de cambio de afiliación puede ser en una implementación del algoritmo de Paxos (Lamport, L.: The Part-Time Parliament, ACM Transactions on Computer Systems 16, 2 (Mayo de 1998), 133-169) donde el conjunto de proponentes (procesos que realizan propuestas a la afiliación), aceptores (procesos que votan sobre si una propuesta debiera acordarse por la afiliación) y aprendedores (procesos en la afiliación que aprenden de acuerdos que se han realizado) pueden rotarse o cambiarse para manejar situaciones donde se eliminan procesos del sistema a medida que se llevan fuera de línea permanentemente y dejan de estar en servicio, o a medida que se añaden nuevos procesos al sistema para conseguir mayor tolerancia a fallos o rendimiento.

La Figura 9 muestra un aspecto de un método de despliegue de una afiliación de nodos en un sistema informático distribuido, de acuerdo con una realización. Como se muestra en la misma y de acuerdo con una realización, puede iniciarse una tarea, *Crear_Tarea_de_Afiliación,* por un nodo de iniciación que se ha inducido en una red existente de nodos (es decir, uno o más nodos de cálculo distribuidos) y esta tarea puede usarse para crear una afiliación del conjunto de nodos de cálculo de los cuales tiene conocimiento el nodo de iniciación. De acuerdo con una realización, esto puede conseguirse usando el siguiente procedimiento:

Papeles:

De acuerdo con una realización, *Crear_Tarea_de_Afiliación* 902 puede configurarse para tener dos papeles: el papel *Creador_de_Afiliación* 906, o el papel supuesto por un nodo que inicia la creación de una nueva afiliación, y el papel *Objetivo_de_Afiliación* 906, o el papel supuesto por un nodo que ha de ser parte de una nueva afiliación, como se especifica por el *Creador_de_Afiliación.* Obsérvese que puede haber uno o más nodos en el *Objetivo_de_Afiliación* definido en *Crear_Tarea_de_Afiliación* 902 y que el *Creador_de_Afiliación* 906 puede estar o no entre los nodos enumerados en los nodos en *Objetivo_de_Afiliación 908,* permitiendo de esta manera que se creen las afiliaciones 'remotas', o que se envíen a un conjunto de otros nodos.

Protocolo:

De acuerdo con una realización, y como se muestra en la Figura 9,

- Como se muestra B91, el nodo que asume el papel de *Creador_de_Afiliación* 906 puede estar configurado para crear *Crear_Tarea_de_Afiliación* 902 con un identificador único *Identidad_de_Tarea* 904.
- El nodo que asume el papel de *Creador_de_Afiliación* puede estar configurado para seleccionar todos o un subconjunto de nodos del conjunto completo de nodos de los cuales tiene conocimiento para formar la afiliación como se muestra en B92 y puede estar configurado para añadir el nodo o nodos seleccionados a *Crear_Tarea_de_Afiliación* 902. Como se ha descrito anteriormente, este subconjunto puede incluir o no el nodo *Creador_de_Afiliación* mismo, permitiendo por lo tanto que se forme una afiliación 'remota' que no incluye el nodo que asume el papel de *Creador_de_Afiliación* 906.
- El nodo que asume el papel de *Creador_de_Afiliación* 906 puede estar configurado adicionalmente para crear una nueva afiliación del conjunto de nodos, para asignar papeles específicos al mismo, como se muestra en B93 y para asociar/añadir la afiliación a *Crear_Tarea_de_Afiliación 902.*
- El inicio de *Crear_Tarea_de_Afiliación* 902 puede hacerse persistir en un almacén persistente 903 e iniciarse, como se muestra en B94.
- La iniciación de la tarea *Crear_Tarea_de_Afiliación* provoca que el nodo de iniciación cree una baliza (una baliza es un proceso configurado para difundir repetitivamente un mensaje a una lista predeterminada de receptores objetivo, eliminar receptores objetivo de la lista predeterminada a la que se difunde el mensaje hasta que se haya recibido un acuse de recibo de respuesta de cada uno de los receptores objetivo), como se muestra en B95, usando la lista de ubicaciones (direcciones) que se ha proporcionado. La baliza creada así puede estar

configurada para enviar, en B96, un mensaje *Crear_Afiliación* a estas direcciones. De acuerdo con una realización, este mensaje *Crear_Afiliación* puede estar configurado para comprender:
- la identidad de *Crear_Tarea_de_Afiliación* 912;
- La nueva afiliación a desplegarse 914; y
- el nodo, ubicación, la identidad, el nombre de anfitrión y/o puerto del nodo que ha iniciado la tarea de *Creador_de_Afiliación*, como se muestra en 916.

La Figura 10 muestra un aspecto adicional de un método de despliegue de una afiliación de nodos en un sistema informático distribuido, de acuerdo con una realización. Como se muestra y de acuerdo con una realización, cuando el nodo *Objetivo_de_Afiliación* 1002 recibe el mensaje *Crear_Afiliación* 1004 del nodo de iniciación (el nodo *Creador_de_Afiliación*), el nodo *Objetivo_de_Afiliación* 1002 puede estar configurado para:

- extraer la afiliación del mensaje, como se muestra en B1002;
- desplegar la afiliación en B1004;
- generar y enviar un mensaje *Crear_Respuesta_de_Afiliación*, como se muestra en B1006 al nodo *Creador_de_Afiliación* 1006 para indicar que se ha desplegado la nueva afiliación. De acuerdo con una realización, el mensaje *Crear_Respuesta_de_Afiliación* puede estar configurado para comprender:
- la identidad de *Crear_Tarea_de_Afiliación* (por ejemplo, el identificador único *Identidad_de_Tarea* 1008; y
- el nodo e identidad de ubicación del *Objetivo_de_Afiliación* que envía el mensaje, como se muestra en 1010.

De acuerdo con una realización, cuando el nodo *Creador_de_Afiliación* 1006 recibe un mensaje *Crear_Respuesta_de_Afiliación*, el nodo *Creador_de_Afiliación* puede:

- extraer la identidad de ubicación del *Objetivo_de_Afiliación* (el nodo de envío) del mensaje, como se muestra en B1008;
- eliminar la identidad de ubicación de la lista de direcciones en la baliza de *Crear_Tarea_de_Afiliación,* como se denomina para en B1010;
- comprobar si hay alguna ubicación especificada en el *Objetivo_de_Afiliación* desde la cual el nodo *Creador_de_Afiliación* no ha escuchado aún (es decir, si el número de *Objetivos_de_Afiliación* aún responde con un mensaje *Crear_Respuesta_de_Afiliación* es mayor que cero), como se muestra en B1012.

Si hay más ubicaciones que no han de responder aún al mensaje *Crear_Afiliación* (hay más de cero ubicaciones que pertenecen en la baliza) el nodo *Creador_de_Afiliación* puede esperar mensajes adicionales (por ejemplo, mensajes *Crear_Respuesta_de_Afiliación* de nodos enumerados en el *Objetivo_de_Afiliación*) a recibirse (NO ramal de B1012). Si, sin embargo, ya no hay más ubicaciones que hayan de responder (no quedan cero ubicaciones en la baliza), se sigue el ramal Sí de B 1012 y el nodo *Creador_de_Afiliación* puede estar configurado para desplegar la misma afiliación/localmente como se muestra en B1014 y hacer persistir el estado de *Crear_Tarea_de_Afiliación,* como se muestra en B1016. La tarea *Crear_Tarea_de_Afiliación* puede marcarse a continuación según se completa, como se muestra en B1018.

Propiedades de seguridad y ejecución

Cualquier protocolo de afiliación, de acuerdo con una realización, puede comprender tanto propiedades de seguridad como de ejecución. De hecho, un protocolo de afiliación de acuerdo con una realización puede proporcionarse con un conjunto de propiedades de seguridad para asegurar que no ocurra nada "malo" durante el despliegue de una nueva afiliación, así como con un conjunto de propiedades de ejecución, para proporcionar la ocurrencia de que está teniendo lugar algo eventualmente "bueno" (por ejemplo, la nueva afiliación se desplegará).

Propiedades de seguridad

De acuerdo con una realización, el despliegue de afiliación es idempotente, en que producirá el mismo resultado si se ejecuta una vez o múltiples veces. Por lo tanto, si el *Creador_de_Afiliación* falla antes de que haya recibido respuestas de todos los nodos enumerados en el *Objetivo_de_Afiliación,* ese nodo puede reiniciar *Crear_Tarea_de_Afiliación* desde el comienzo y reenviar mensajes *Crear_Afiliación* duplicados a cada uno de los nodos objetivo sin cambiar el resultado eventual. Tras la recepción del mensaje *Crear_Afiliación,* los nodos objetivo pueden responder con *Crear_Respuesta_de_Afiliación* que indica que se ha desplegado la nueva afiliación. No hay necesidad de que *Crear_Tarea_de_Afiliación* haya de hacerse persistir cada vez que se reciba un mensaje *Crear_Respuesta_de_Afiliación*: la única vez que *Crear_Tarea_de_Afiliación* necesita hacerse persistir (distinto de cuando se crea en primer lugar) es cuando todas las respuestas se han recibido y la tarea puede ser/marcarse como que se ha completado.

Propiedades de ejecución

El uso de una baliza, de acuerdo con una realización, asegura que al menos un mensaje *Crear_Afiliación* alcanzará eventualmente el nodo dirigido en el *Objetivo_de_Afiliación.* Si se pierde la respuesta a este mensaje (es decir, el mensaje *Crear_Respuesta_de_Afiliación*) también se pierde, el *Creador_de_Afiliación* puede mantener el envío del

mensaje *Crear_Afiliación* (siendo idempotente) hasta que reciba *Crear_Respuesta_de_Afiliación* anticipada de los nodos objetivo. Como dictan las propiedades de seguridad que la recepción de múltiples mensajes *Crear_Afiliación* es idempotente, esta combinación asegurará que eventualmente se complete el proceso.

La Figura 11 ilustra un diagrama de bloques de un sistema informático 1100 en el que pueden implementarse las realizaciones. El sistema informático 1100 puede incluir un bus 1101 u otro mecanismo de comunicación para comunicar información, y uno o más procesadores 1102 acoplados con el bus 1101 para procesar información. El sistema informático 1100 comprende adicionalmente una memoria de acceso aleatorio (RAM) u otro dispositivo de almacenamiento dinámico 1104 (denominado como memoria principal), acoplado al bus 1101 para almacenar información e instrucciones a ejecutarse por los procesadores) 1102. La memoria principal 1104 también puede usarse para almacenar variables temporales u otra información intermedia durante la ejecución de instrucciones por el procesador 1102. El sistema informático 1100 puede incluir también una memoria de solo lectura (ROM) y/u otro dispositivo de almacenamiento estático 1106 acoplado al bus 1101 para almacenar información estática e instrucciones para el procesador 1102. Un dispositivo de almacenamiento de datos 1107 tal como, por ejemplo, un disco magnético o almacenamiento Flash, puede acoplarse al bus 1101 para almacenar información e instrucciones. El sistema informático 1100 puede también estar acoplado mediante el bus 1101 a un dispositivo de visualización 1110 para visualizar información a un usuario informático. Un dispositivo de entrada alfanumérico 1122, que incluye teclas alfanuméricas y otras, puede estar acoplado al bus 1101 para comunicar información y selecciones de comando a procesadores) 1102. Otro tipo de dispositivo de entrada de usuario es control de cursor 1123, tal como un ratón, una bola de mando o teclas de dirección de cursor para comunicar información de dirección y ordenar selecciones al procesador 1102 y para controlar movimiento de cursor en el visualizador 1121. El sistema informático 1100 puede estar acoplado, mediante un dispositivo de comunicación (por ejemplo, modem, NIC) a una red 1126 y a uno o más nodos de un sistema informático distribuido.

Las realizaciones están relacionadas con el uso del sistema informático y/o con una pluralidad de tales sistemas informáticos para crear y desplegar afiliaciones en un sistema informático distribuido. De acuerdo con una realización, los métodos y sistemas descritos en el presente documento pueden proporcionarse por uno o más sistemas informáticos 1100 en respuesta al procesador o procesadores 1102 que ejecutan secuencias de instrucciones contenidas en la memoria 1104. Tales instrucciones pueden leerse en memoria 1104 de otro medio legible por ordenador, tal como un dispositivo de almacenamiento de datos 1107. La ejecución de la secuencias de instrucciones contenidas en memoria 1104 provoca que el procesador o procesadores 1102 realicen las etapas y tengan la funcionalidad descrita en el presente documento. En realizaciones alternativas, puede usarse circuitería de cableado permanente en lugar de, o en combinación, con instrucciones de software para implementar la presente invención. Por lo tanto, la presente invención no está limitada a combinación específica alguna de circuitería de hardware y software. De hecho, debería entenderse por los expertos en la materia que cualquier sistema informático adecuado puede implementar la funcionalidad descrita en el presente documento. El sistema informático puede incluir uno o una pluralidad de microprocesadores que funcionan para realizar las funciones deseadas. En una realización, las instrucciones ejecutadas por el microprocesador o microprocesadores son operables para provocar que el microprocesador o microprocesadores realicen las etapas descritas en el presente documento. Las instrucciones pueden almacenarse en cualquier medio legible por ordenador. En una realización, pueden almacenarse en una memoria de semiconductores no volátil externa al microprocesador, o integrarse con el microprocesador. En otra realización, las instrucciones pueden almacenarse en un disco y leerse en una memoria de semiconductores volátil antes de su ejecución por el microprocesador.

Aunque se han descrito ciertas realizaciones de las invenciones, estas realizaciones se han presentado a modo de ejemplo únicamente, y no se pretende que limiten el alcance de las invenciones. De hecho, los métodos novedosos, dispositivos y sistemas descritos en el presente documento pueden realizarse en diversas otras formas. Adicionalmente, pueden realizarse diversas omisiones, sustituciones y cambios en forma de los métodos y sistemas descritos en el presente documento sin alejarse de las invenciones. Las reivindicaciones adjuntas y sus equivalentes se pretende que cubran tales formas o modificaciones como que caerían dentro del alcance de las invenciones. Por ejemplo, los expertos en la materia apreciarán que en diversas realizaciones, las estructuras reales (tales como, por ejemplo,) pueden diferir de aquellas mostradas en las figuras. Dependiendo de la realización, pueden eliminarse ciertas de las etapas descritas en el ejemplo anterior, otras pueden añadirse. También, las características y atributos de las realizaciones específicas anteriormente desveladas pueden combinarse en diferentes maneras para formar realizaciones adicionales, todas las cuales caen dentro del alcance de la presente divulgación. Aunque la presente divulgación proporciona ciertas realizaciones y aplicaciones preferidas, otras realizaciones que son evidentes para los expertos en la materia, que incluyen realizaciones que no proporcionan todas las características y ventajas expuestas en el presente documento, también se encuentran dentro del alcance de esta divulgación. Por consiguiente, el alcance de la presente divulgación se pretende que esté definido únicamente por referencia a las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un método implementado por ordenador de despliegue de una afiliación de nodos en un sistema informático distribuido (100), comprendiendo el método implementado por ordenador;
seleccionar (B92) nodos (115), para que sean parte de la afiliación de nodos a desplegarse;
crear (B94) una tarea de afiliación, que identifica un nodo creador de afiliación (906) como el nodo que está creando la afiliación a desplegarse, y que comprende un objetivo de afiliación (908), que identifica una pluralidad de nodos objetivo del sistema informático distribuido (100), que han de convertirse en miembros de la afiliación; y
crear (B95) una baliza, configurada para difundir repetitivamente, a través de una red informática (1126), un mensaje de creación de afiliación (B96) a cada nodo objetivo identificado en el objetivo de afiliación, comprendiendo el mensaje de creación de afiliación (B96) al menos una identidad (912) de la tarea de afiliación y una identificación (914) de la afiliación a desplegarse;
después de recibir (B1008) el nodo creador de afiliación una respuesta de un nodo objetivo en el objetivo de afiliación:

eliminar (B1010) el nodo objetivo, a partir del cual se recibió la respuesta de la baliza, de manera que la baliza ya no difunde más el mensaje de creación de afiliación al mismo; y
desplegar (B1014) la afiliación, cuando se ha recibido una respuesta de cada uno de los nodos objetivo identificados en el objetivo de afiliación; y
si el nodo creador de afiliación falla antes de que haya recibido respuestas de cada uno de los nodos objetivo idesntificados en el objetivo de afiliación, reiniciar el nodo creador de afiliación la tarea de afiliación y reenviar mensajes de creación de afiliación duplicados a cada uno de los nodos objetivo.

2. El método implementado por ordenador de la reivindicación 1, que comprende adicionalmente asignar al menos un papel a los nodos seleccionados para que sean parte de la afiliación a desplegarse.

3. El método implementado por ordenador de la reivindicación 2, en el que el al menos un papel comprende un papel de un creador de afiliación y un papel de un objetivo de afiliación.

4. El método implementado por ordenador de las reivindicaciones 1, 2 o 3, que comprende adicionalmente hacer persistir la tarea de afiliación creada en un almacén persistente (903).

5. El método implementado por ordenador de cualquier reivindicación anterior, en el que el mensaje de creación de afiliación comprende adicionalmente una identificación del nodo creador de afiliación y que posibilita información para posibilitar la comunicación con el nodo creador de afiliación, comprendiendo la información que posibilita al menos uno de una identificación, una ubicación y un nombre de anfitrión del nodo creador de afiliación.

6. El método implementado por ordenador de cualquier reivindicación anterior, que comprende adicionalmente, después de haber desplegado la afiliación, hacer persistir un estado de la tarea de afiliación creada.

7. Un dispositivo informático (1100), que comprende:

una memoria (1104, 1106);
un procesador (1102), dispuesto para engendrar una pluralidad de procesos, estando configurados los procesos para provocar que el dispositivo informático (1100) despliegue una afiliación de nodos en un sistema informático distribuido (100), comprendiendo la pluralidad de procesos lógica de procesamiento para:

seleccionar (B92) nodos (115) para que sean parte de la afiliación de nodos a desplegarse;
crear (B94) una tarea de afiliación, que identifica un nodo creador de afiliación (906) como el nodo que está creando la afiliación a desplegarse, y que comprende un objetivo de afiliación (908), que identifica una pluralidad de nodos del sistema informático distribuido (100), que han de convertirse en miembros de la afiliación;
crear (B95) una baliza configurada para difundir repetitivamente, a través de una red informática (1126), un mensaje de creación de afiliación (B96) a cada nodo objetivo identificado en el objetivo de afiliación, comprendiendo el mensaje de creación de afiliación (B96) al menos una identidad (912) de la tarea de afiliación y una identificación (914) de la afiliación a desplegarse; y
tras recibir (B1008) una respuesta de un nodo objetivo en el objetivo de afiliación:

eliminar (B1010) el nodo objetivo, a partir del cual se recibió la respuesta de la baliza, de manera que la baliza ya no difunde más el mensaje de creación de afiliación al mismo; y
desplegar (B1014) la afiliación, cuando se ha recibido una respuesta de cada uno de los nodos objetivo identificados en el objetivo de afiliación; y

si el nodo creador de afiliación falla antes de que haya recibido respuestas de cada uno de los nodos objetivo identificados en el objetivo de afiliación,
reiniciar, por el nodo creador de afiliación, la tarea de afiliación y reenviar, por el nodo creador de afiliación,

mensajes de creación de afiliación duplicados a cada uno de los nodos objetivo.

8. El dispositivo informático de la reivindicación 7, que comprende adicionalmente lógica de procesamiento dispuesta para asignar al menos un papel a los nodos seleccionados para que sean parte de la afiliación a desplegarse.

9. El dispositivo informático de las reivindicaciones 7 u 8, que comprende adicionalmente lógica de procesamiento dispuesta para hacer persistir la tarea de afiliación creada en un almacén persistente (903).

10. El dispositivo informático de cualquiera de las reivindicaciones 7 a 9, que comprende adicionalmente lógica de procesamiento dispuesta para, después de haber desplegado la afiliación, hacer persistir un estado de la tarea de afiliación creada.

11. El método de cualquiera de las reivindicaciones 1 a 6, o el dispositivo informático de cualquiera de las reivindicaciones 7 a 10, en donde la tarea de afiliación es identificada por un identificador de tarea único (1008).

12. El método de cualquiera de las reivindicaciones 1 a 6, o el dispositivo informático de cualquiera de las reivindicaciones 7 a 10, en donde el despliegue de la afiliación es idempotente.

13. Un medio legible por máquina tangible que tiene datos, almacenados en el mismo, que representan secuencias de instrucciones que, cuando se ejecutan por un dispositivo informático (1100), provocan que el dispositivo informático despliegue una afiliación de nodos en un sistema informático distribuido (100), realizando la secuencia de instrucciones las etapas de método de cualquiera de las reivindicaciones 1 a 6.

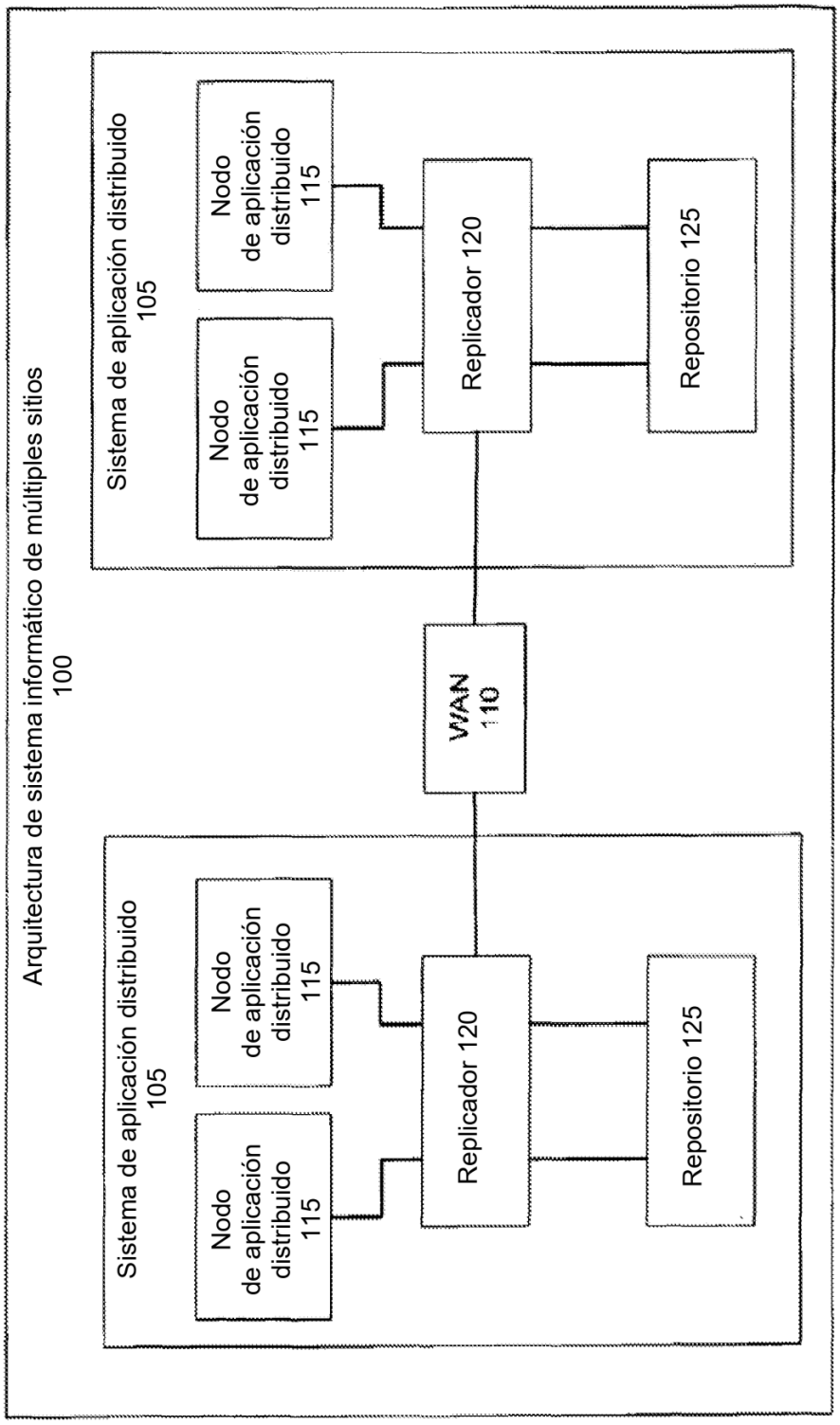
Arquitectura de sistema informático de múltiples sitios
100
Sistema de aplicación distribuido
105
Nodo de aplicación distribuido 115
Nodo de aplicación distribuido 115
Replicador 120
Repositorio 125
WAN
110
Sistema de aplicación distribuido
105
Nodo de aplicación distribuido 115
Nodo de aplicación distribuido 115
Replicador 120
Repositorio 125

FIG. 1

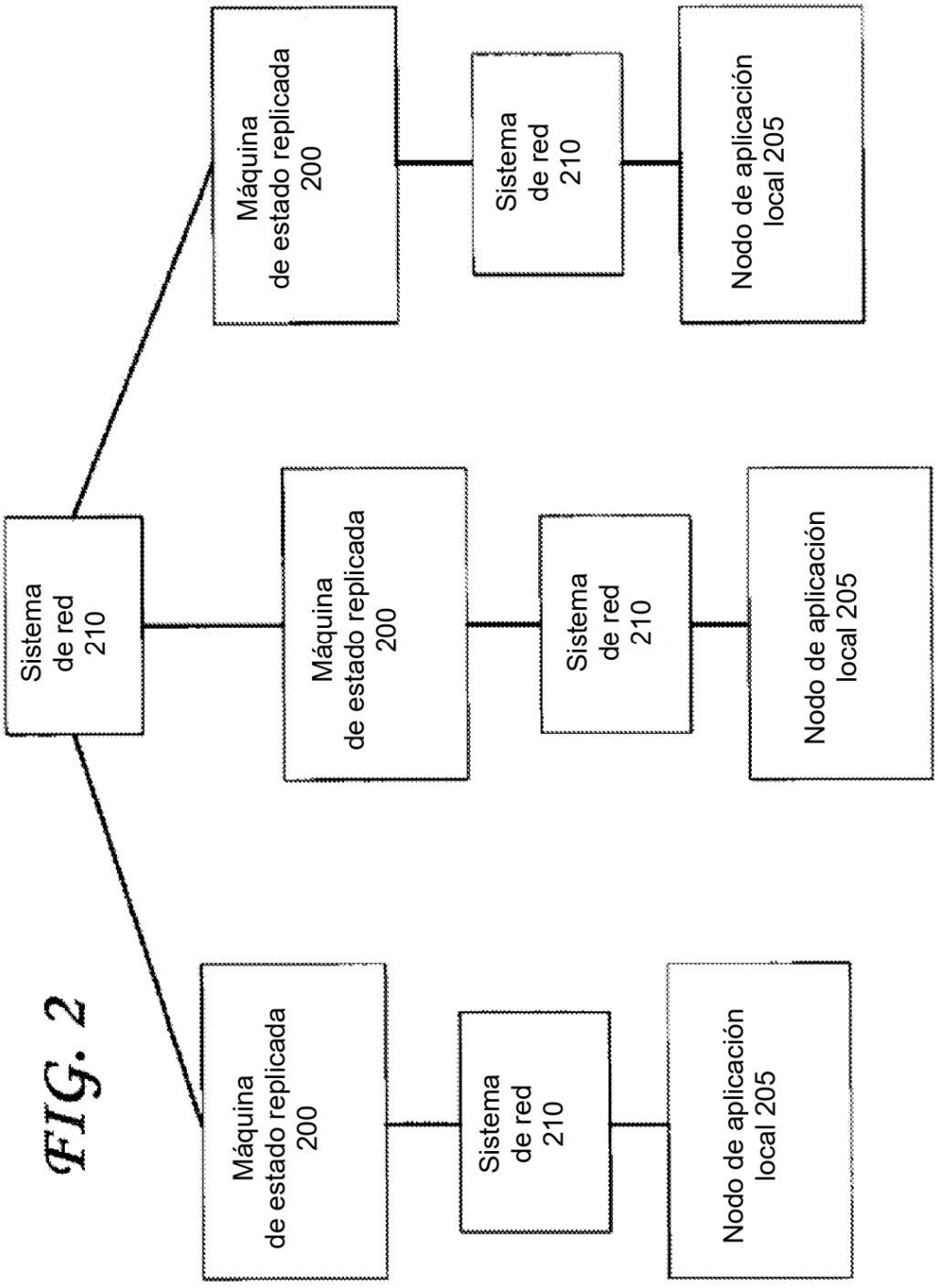
Sistema de red 210
Máquina de estado replicada 200
Sistema de red 210
Nodo de aplicación local 205
Máquina de estado replicada 200
Sistema de red 210
Nodo de aplicación local 205
Máquina de estado replicada 200
Sistema de red 210
Nodo de aplicación local 205

FIG. 2

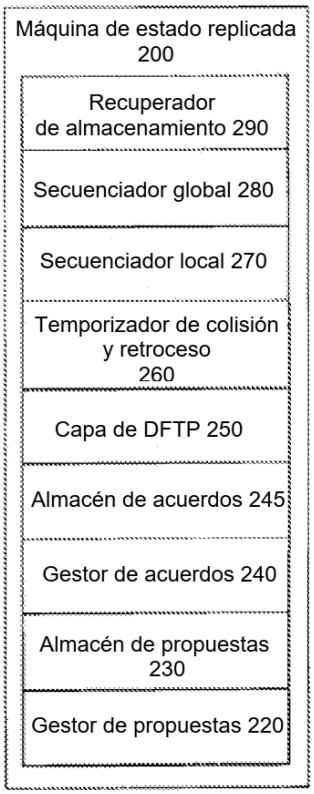
Máquina de estado replicada 200
Recuperador de almacenamiento 290
Secuenciador global 280
Secuenciador local 270
Temporizador de colisión y retroceso 260
Capa de DFTP 250
Almacén de acuerdos 245
Gestor de acuerdos 240
Almacén de propuestas 230
Gestor de propuestas 220

FIG. 3

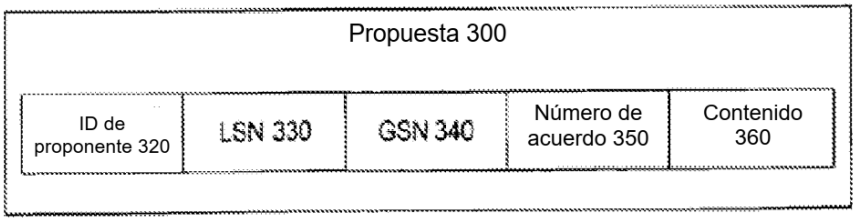
Propuesta 300
ID de proponente 320
LSN 330
GSN 340
Número de acuerdo 350
Contenido 360

FIG. 4

Secuencia local
400

Proponente ID=0x123	LSN	GSN	Número de acuerdo	Contenido
Proponente ID=0x123	LSN	GSN	Número de acuerdo	Contenido
Proponente ID=0x123	LSN	GSN	Número de acuerdo	Contenido

FIG. 5

Secuencia global
500

GSN n.º 1	Manejador de secuencia local 400
GSN n.º 2	Manejador de secuencia local 400
GSN n.º 3	Manejador de secuencia local 400
GSN n.º 4	Manejador de secuencia local 400

FIG. 6

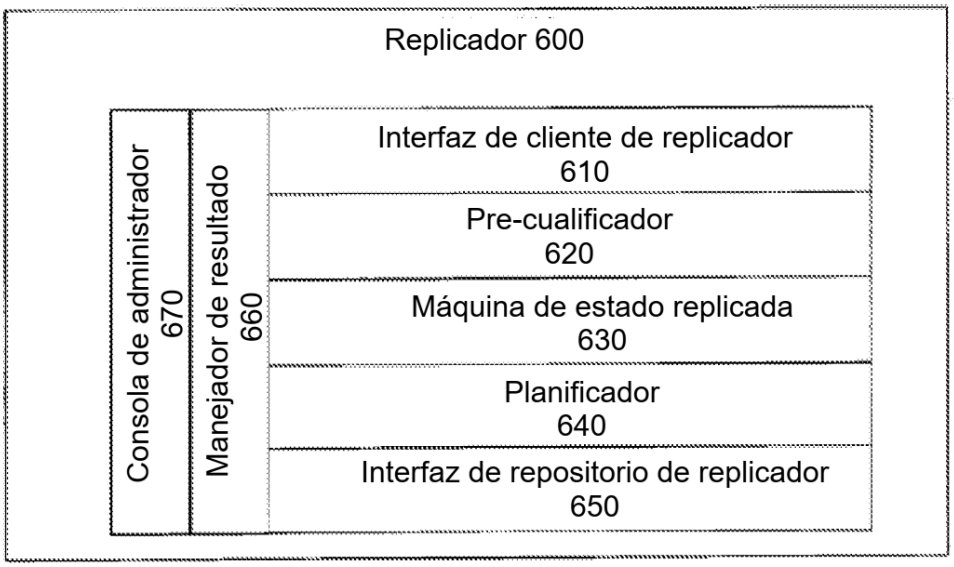
Replicador 600
Consola de administrador 670
Manejador de resultado 660
Interfaz de cliente de replicador 610
Pre-cualificador 620
Máquina de estado replicada 630
Planificador 640
Interfaz de repositorio de replicador 650

FIG. 7

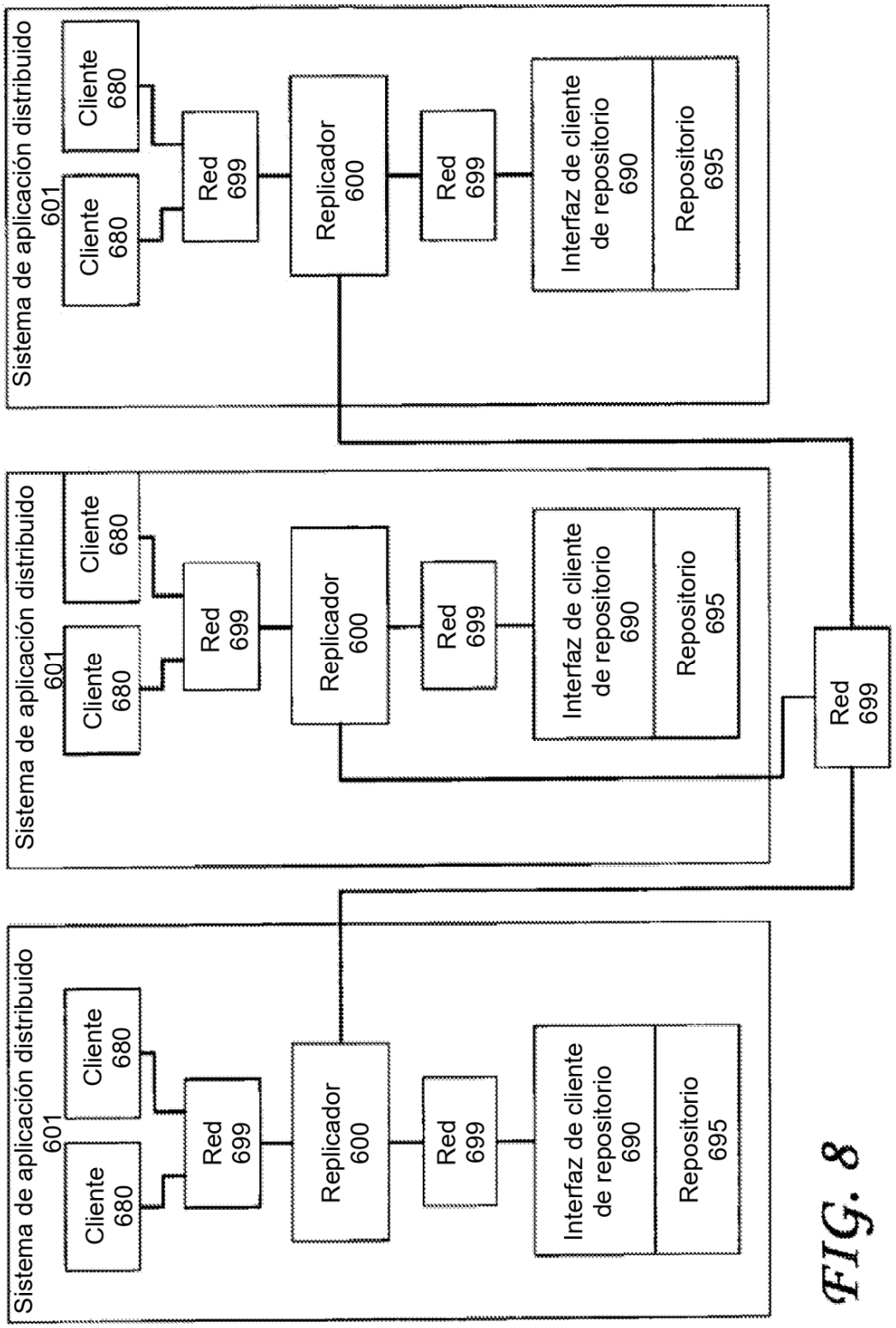
Sistema de aplicación distribuido
601
Cliente 680
Cliente 680
Red 699
Replicador 600
Red 699
Interfaz de cliente de repositorio 690
Repositorio 695
Sistema de aplicación distribuido
601
Cliente 680
Cliente 680
Red 699
Replicador 600
Red 699
Interfaz de cliente de repositorio 690
Repositorio 695
Red 699
Sistema de aplicación distribuido
601
Cliente 680
Cliente 680
Red 699
Replicador 600
Red 699
Interfaz de cliente de repositorio 690
Repositorio 695

FIG. 8

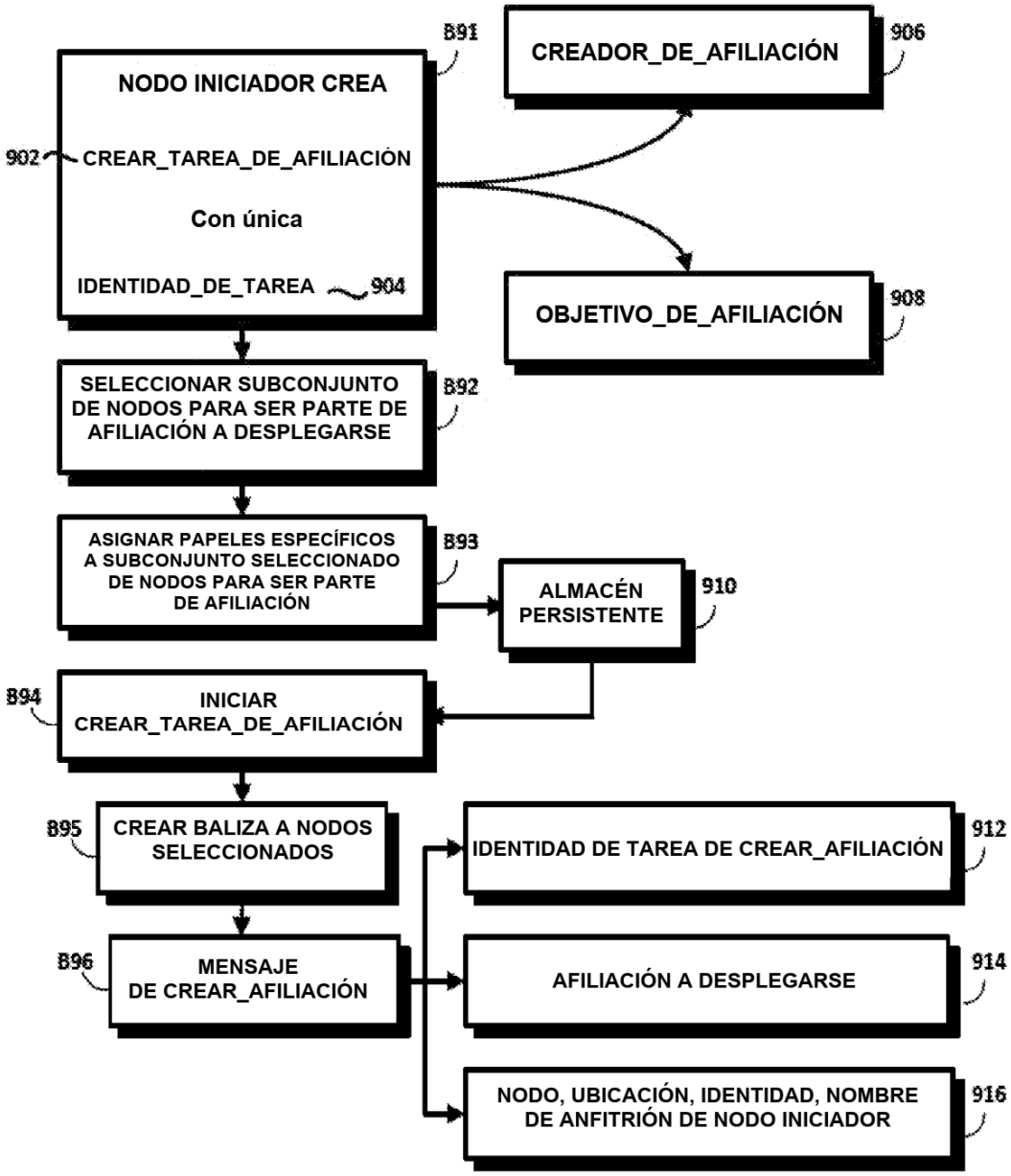
891
NODO INICIADOR CREA
902
CREAR_TAREA_DE_AFILIACIÓN
Con única
IDENTIDAD_DE_TAREA
904
906
CREADOR_DE_AFILIACIÓN
908
OBJETIVO_DE_AFILIACIÓN
892
SELECCIONAR SUBCONJUNTO DE NODOS PARA SER PARTE DE AFILIACIÓN A DESPLEGARSE
893
ASIGNAR PAPELES ESPECÍFICOS A SUBCONJUNTO SELECCIONADO DE NODOS PARA SER PARTE DE AFILIACIÓN
910
ALMACÉN PERSISTENTE
894
INICIAR CREAR_TAREA_DE_AFILIACIÓN
895
CREAR BALIZA A NODOS SELECCIONADOS
896
MENSAJE DE CREAR_AFILIACIÓN
912
IDENTIDAD DE TAREA DE CREAR_AFILIACIÓN
914
AFILIACIÓN A DESPLEGARSE
916
NODO, UBICACIÓN, IDENTIDAD, NOMBRE DE ANFITRIÓN DE NODO INICIADOR

FIG. 9

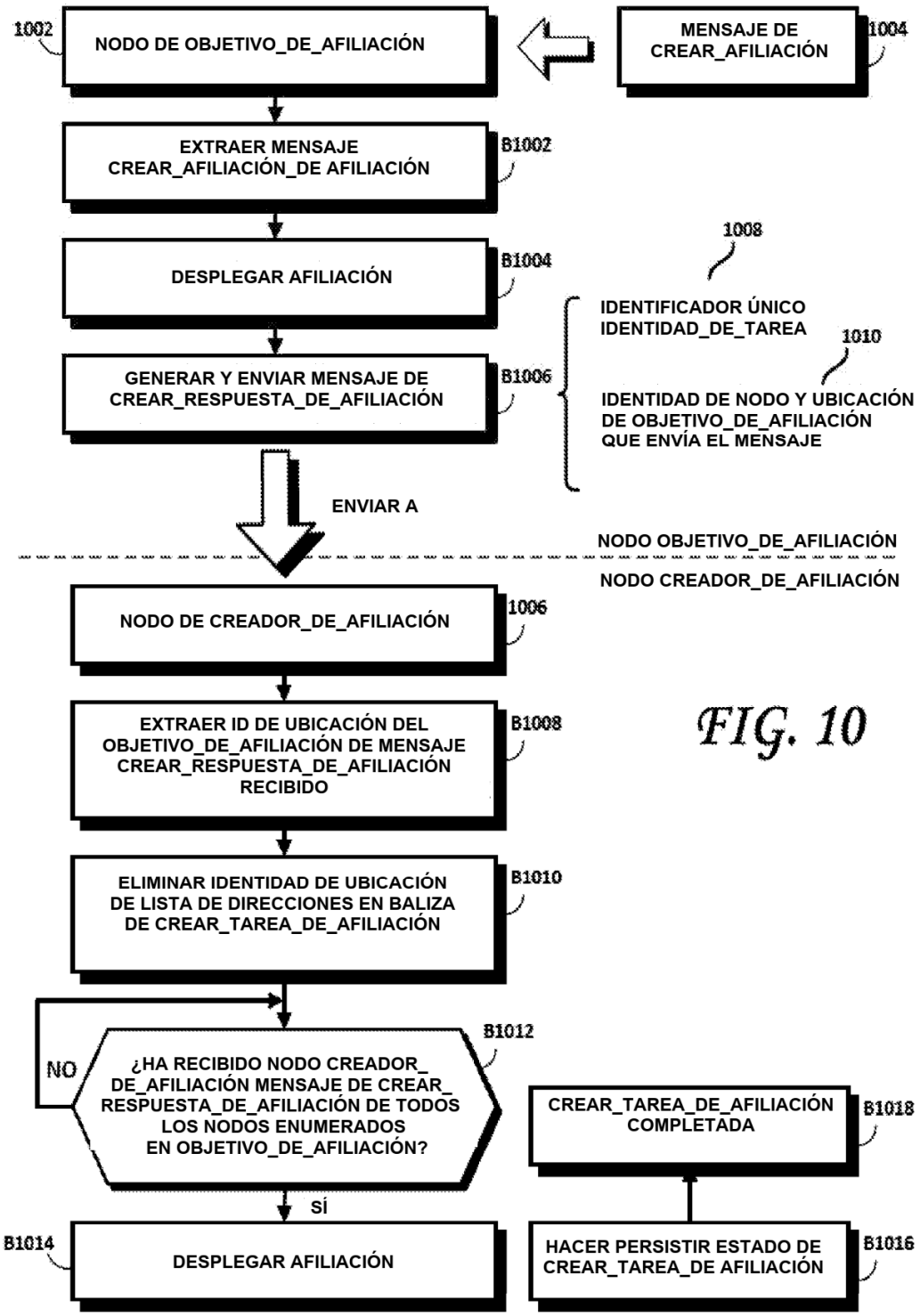
1002
NODO DE OBJETIVO_DE_AFILIACIÓN
MENSAJE DE CREAR_AFILIACIÓN
1004
EXTRAER MENSAJE CREAR_AFILIACIÓN_DE AFILIACIÓN
B1002
DESPLEGAR AFILIACIÓN
B1004
1008
IDENTIFICADOR ÚNICO IDENTIDAD_DE_TAREA
1010
GENERAR Y ENVIAR MENSAJE DE CREAR_RESPUESTA_DE_AFILIACIÓN
B1006
IDENTIDAD DE NODO Y UBICACIÓN DE OBJETIVO_DE_AFILIACIÓN QUE ENVÍA EL MENSAJE
ENVIAR A
NODO OBJETIVO_DE_AFILIACIÓN
NODO CREADOR_DE_AFILIACIÓN
NODO DE CREADOR_DE_AFILIACIÓN
1006
EXTRAER ID DE UBICACIÓN DEL OBJETIVO_DE_AFILIACIÓN DE MENSAJE CREAR_RESPUESTA_DE_AFILIACIÓN RECIBIDO
B1008
ELIMINAR IDENTIDAD DE UBICACIÓN DE LISTA DE DIRECCIONES EN BALIZA DE CREAR_TAREA_DE_AFILIACIÓN
B1010
NO
¿HA RECIBIDO NODO CREADOR_ DE_AFILIACIÓN MENSAJE DE CREAR_ RESPUESTA_DE_AFILIACIÓN DE TODOS LOS NODOS ENUMERADOS EN OBJETIVO_DE_AFILIACIÓN?
B1012
SÍ
CREAR_TAREA_DE_AFILIACIÓN COMPLETADA
B1018
B1014
DESPLEGAR AFILIACIÓN
HACER PERSISTIR ESTADO DE CREAR_TAREA_DE_AFILIACIÓN
B1016

FIG. 10

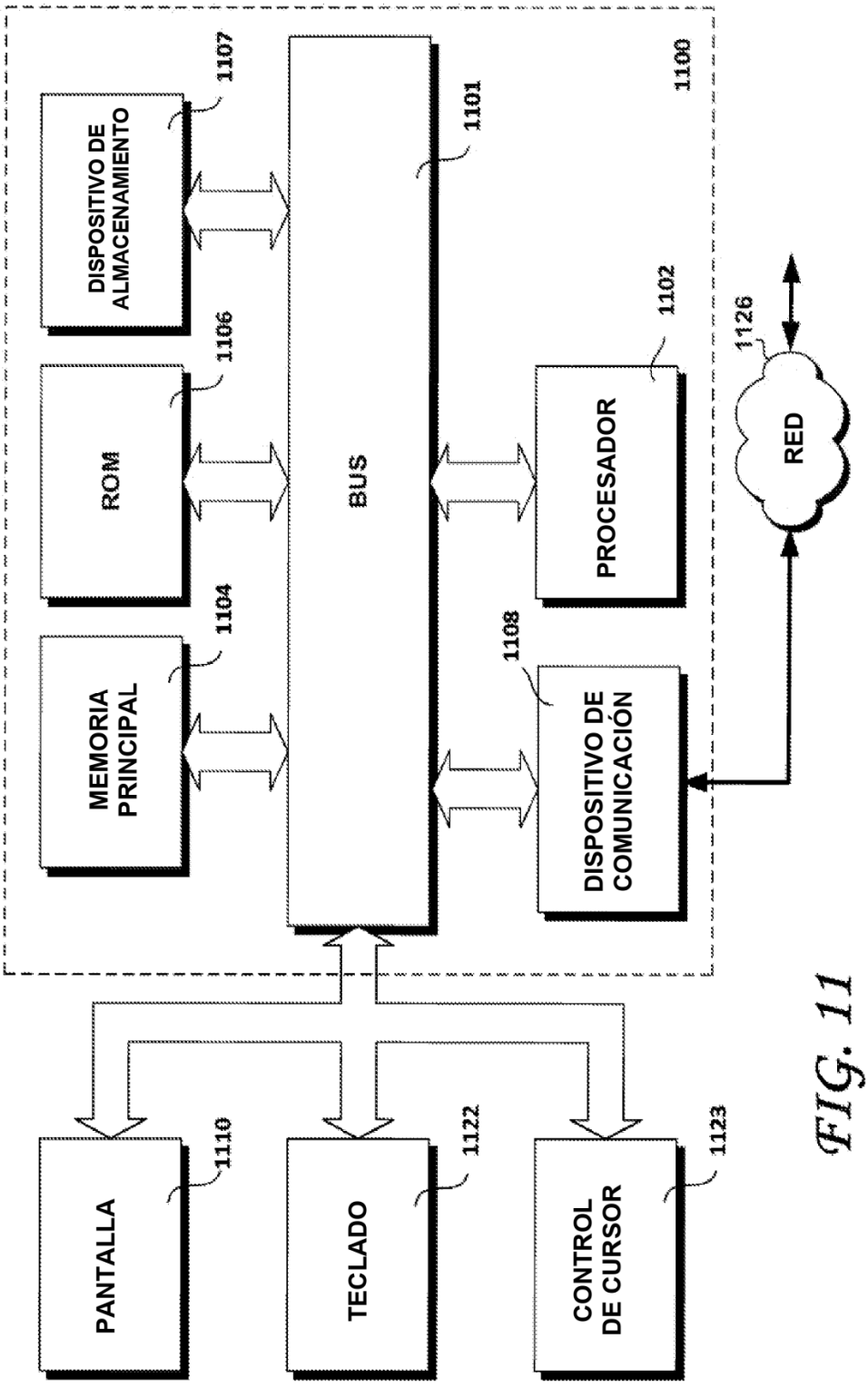
1100
1101
1102
1104
1106
1107
1108
1110
1122
1123
1126
DISPOSITIVO DE ALMACENAMIENTO
ROM
MEMORIA PRINCIPAL
BUS
PROCESADOR
DISPOSITIVO DE COMUNICACIÓN
RED
PANTALLA
TECLADO
CONTROL DE CURSOR

FIG. 11