(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 811 106**

(51) Int. CI.:

G06F 16/27 (2009.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **18.03.2011 PCT/US2011/029056**

(87) Fecha y número de publicación internacional: **22.09.2011 WO11116324**

(96) Fecha de presentación y número de la solicitud europea: **18.03.2011 E 11712105 (3)**

(97) Fecha y número de publicación de la concesión europea: **13.05.2020 EP 2548135**

(54) Título: **Sistema de gestión de base de datos**

(30) Prioridad:

18.03.2010 US 315351 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.03.2021

(73) Titular/es:

NUODB INC. (100.0%)
215 First Street, Suite 005
Cambridge, MA 02142, US

(72) Inventor/es:

STARKEY, JAMES, A.

(74) Agente/Representante:

ZUAZO ARALUZE, Alexander

DESCRIPCIÓN

Sistema de gestión de base de datos

Campo técnico

Esta invención se refiere, en general, a sistemas de gestión de base de datos. Más específicamente, esta invención se refiere a un método y un aparato para implementar una atomicidad, un rendimiento y una escalabilidad caracterizados de sistema de gestión de base de datos relacional distribuida, multiusuario, elásticos, bajo demanda.

Antecedentes de la técnica

Durante los últimos años, se ha demostrado el uso de bases de datos para almacenar y recuperar mensajes como una herramienta importante en una amplia variedad de aplicaciones comerciales. Inicialmente, muchos sistemas de bases de datos funcionaban en una instalación de un único servidor con múltiples usuarios. Sin embargo, se han desarrollado diversos factores durante los últimos años que han requerido que cambie la naturaleza básica de la arquitectura de bases de datos. Como primer factor, los requisitos de almacenamiento de bases de datos se han vuelto extremadamente grandes. En segundo lugar, el número de usuarios que intentan acceder a tales bases de datos también se ha vuelto grande. En tercer lugar, el uso de bases de datos para recuperar datos relativamente estables con mínimas actualizaciones se ha sustituido por el procesamiento de transacciones.

Una transacción es una unidad de trabajo que debe completarse en su totalidad. Una única transacción puede incluir múltiples manipulaciones de datos. Como ejemplo, una única transacción puede incluir una operación de lectura seguida por una operación de escritura. En los últimos años, se ha dirigido un esfuerzo significativo a facilitar que las bases de datos relacionales soporten velocidades de procesamiento de transacciones en continuo aumento.

Ahora, las bases de datos se evalúan mediante una norma que define propiedades de ACID, concretamente: atomicidad, consistencia, aislamiento y durabilidad. La atomicidad garantiza que todas las tareas de transacción se completarán en su totalidad. La consistencia garantiza que sólo se escriben datos válidos en la base de datos. El aislamiento garantiza que otras operaciones no pueden acceder o “ver” datos en un estado intermedio durante una transacción. La durabilidad garantiza que una vez que una transacción se ha procesado satisfactoriamente, no puede anularse.

La consistencia es importante concretamente en sistemas multiusuario en los que es posible que dos o más usuarios intenten un acceso simultáneo a datos volátiles compartidos. Los primeros sistemas multiusuario usaban operaciones de bloqueo para garantizar la consistencia. Los bloqueos podían ser bloqueos exclusivos, o de escritura, o bloqueos no exclusivos, o de lectura y podían aplicarse a registros individuales o a páginas. Sin embargo, a medida que las bases de datos han crecido de tamaño y a medida que las velocidades de transacción han aumentado, la tara para gestionar bloqueos se ha vuelto significativa y, en algunos casos, prohibitiva.

El control de concurrencia multiversión (MVCC, *multi-version concurrency control*) es un proceso alternativo para garantizar la concurrencia. El MVCC puede ser más efectivo que los bloqueos con bases de datos complejas. El MVCC usa sellos de tiempo o identificaciones (ID) de transacción crecientes para serializar diferentes versiones de un registro. Cada versión permite que una transacción lea la versión más reciente de un objeto que precede al sello de tiempo o la ID. Con este método de control, cualquier cambio en un registro, por ejemplo, no se verá por otros usuarios hasta que se verifique el cambio. El MVCC también elimina bloqueos con otra tara acompañante y establece un sistema en el que las operaciones de lectura no pueden bloquear las operaciones de escritura.

Además, para cumplir las pruebas de ACID, ahora existe un requisito de disponibilidad continua para los usuarios. Algunos sistemas de bases de datos dedican un sistema informático para el procesamiento de transacciones y otro para el soporte de decisiones y otros procesos de notificación. Están interconectados de modo que pueden soportarse simultáneamente otras funciones. A medida que crecen el tamaño y la complejidad de las bases de datos, los sistemas de procesamiento de datos existentes se sustituyen por un sistema de procesamiento de datos más potente. Otro enfoque para adaptarse al crecimiento implica sistemas reproducidos donde una máquina se designa como una máquina “principal” que mantiene todas las máquinas reproducidas en sincronismo. Si fallara una máquina principal, un proceso asignaría esa función a otra máquina reproducida. Están disponibles diferentes máquinas para determinados usuarios. Este enfoque no puede variarse a escala porque todas las máquinas han de tener la misma capacidad.

Como otro enfoque, múltiples sistemas de bases de datos autónomos pueden integrarse en una única base de datos “federada” con una red informática que interconecta las diversas bases de datos individuales. Las bases de datos federadas requieren “software intermedio” (*middleware*) para mantener las bases de datos constituyentes en sincronismo. Este “software intermedio” puede volverse muy complejo. A medida que aumenta el tamaño de las bases de datos, los recursos requeridos para hacer funcionar el software intermedio pueden imponer una tara tan suficientemente grande que se deteriora el rendimiento global del sistema.

La “creación de particiones“ es otro enfoque para implementar bases de datos en las que una base de datos lógica o sus elementos constituyentes se dividen en distintas partes independientes. En un sistema de gestión de bases de datos distribuido, cada partición puede extenderse por múltiples nodos. En un nodo dado, los usuarios pueden realizar transacciones locales en la partición. La creación de particiones también puede implementarse formando bases de datos más pequeñas o dividiendo elementos seleccionados de tan sólo una tabla.

Hay dos enfoques generales para la creación de particiones. En la creación de particiones horizontal, también denominada “fragmentación”, se colocan diferentes filas en diferentes tablas y diferentes servidores. En general, tienen determinados aspectos en común tales como una gama de códigos postales o apellidos que se dividen en diferentes tablas por intervalos. Por ejemplo, una primera base de datos puede contener todos los registros para apellidos en el intervalo de A M; una segunda base de datos, en el intervalo de N a Z. La fragmentación, que es una forma de creación de particiones horizontal, implica ubicar filas de una base de datos en servidores independientes. La fragmentación sí que reduce el número de filas en cada tabla y aumenta el rendimiento de búsqueda. Sin embargo, la fragmentación usa un código de troceo (*hash*) a nivel de aplicación que hace que la misma sea mucho más difícil de implementar. También incorpora una verificación en dos fases. Las complejidades de la fragmentación hacen que la misma sea adecuada para aplicaciones particulares dado que la base para definir los fragmentos está bastante bien definida.

La creación de particiones vertical implica la creación de tablas con menos columnas y la división de las columnas a lo largo de las tablas. Al igual que una base de datos federada, la creación de particiones vertical requiere software intermedio para determinar cómo encaminar cualquier petición de un campo particular hasta una partición apropiada. Además, estos sistemas funcionan usando una verificación de secuencia en dos fases que es complicada de implementar.

Todavía en otro enfoque, conocido como arquitectura “nada compartido”, cada nodo es independiente y autosuficiente. La arquitectura “nada compartido” es popular para el desarrollo *web* porque puede aumentarse a escala simplemente añadiendo nodos en forma de ordenadores económicos. Este enfoque es popular en aplicaciones de almacenamiento de datos en las que las actualizaciones tienden a ser menos frecuentes de lo que sucedería con el procesamiento de transacciones. Sin embargo, el procesamiento de combinaciones es muy complejo en grandes conjuntos de datos procedentes de diferentes particiones o máquinas.

Algunos sistemas de bases de datos se denominan sistemas “distribuidos”. Una implementación de un sistema distribuido incorpora “agrupaciones” y dos trayectorias de comunicaciones. Una trayectoria de Internet de alta velocidad porta datos entre las agrupaciones. Se requieren trayectorias dedicadas de alta velocidad de comunicaciones para diversas funciones de control, tales como gestión de bloqueo. Aunque este enfoque soluciona los problemas de redundancia y disponibilidad para bases de datos, la gestión de bloqueo, tal como se comentó anteriormente, puede limitar el rendimiento de sistema.

En un sistema “todo compartido”, las comunicaciones de superalta velocidad mantienen el sistema en sincronismo. Sin embargo, la gestión de bloqueo puede requerir recursos de ancho de banda significativos. Para evitar esto, tales sistemas incorporan canales de comunicaciones punto a punto y un controlador de disco muy sofisticado.

De manera colectiva, los sistemas de la técnica anterior satisfacen algunos, pero no todos, de los requisitos conocidos para un sistema de bases de datos. Lo que se requiere es una arquitectura de base de datos que pueda variarse a escala, que cumpla con las propiedades de ACID de atomicidad, consistencia, aislamiento y durabilidad. Lo que también se requiere es un sistema de bases de datos que funcione en Internet sin requerir trayectorias de comunicaciones de alta velocidad dedicadas, que proparte procesamiento de transacciones y que pueda hacerse funcionar en un área geográfica amplia. La publicación “Shared Storage Clusters”, Yousif M., Cluster Computing, Baltzer Science Publishers, vol. 4, n.º 4, 1 de enero de 1999, páginas 247 a 257, da a conocer arquitecturas informáticas de almacenamiento compartido.

Divulgación de la invención

Es un objeto de esta invención proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido.

Otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que es tolerante a fallos.

Todavía otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que tiene un alto grado de disponibilidad.

Aún otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que es independiente de la plataforma.

Todavía aún otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a

escala, bajo demanda, distribuido que es atómico, consistente, aislado y duradero.

Aún todavía otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que funciona en Internet sin requerir trayectorias de comunicaciones de alta velocidad dedicadas.

Todavía aún otro objeto de esta invención es proporcionar un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que proporciona procesamiento de transacciones y que está adaptado para la implementación en un área geográfica amplia.

Según la invención, se proporciona un sistema de gestión de bases de datos según se define en una cualquiera de las reivindicaciones adjuntas. Según una realización, un sistema de gestión de bases de datos que permite que los usuarios interaccionen con una base de datos que se compone de datos y metadatos comprende una pluralidad de nodos y almacenamiento persistente con trayectorias de comunicaciones entre los mismos. Cada nodo incluye una interfaz entre órdenes de entrada y salida de alto nivel a nivel de usuario y órdenes de entrada y salida a nivel del sistema que controlan una secuencia de operaciones para interaccionar con la base de datos, en el que, en respuesta a determinadas órdenes a nivel del sistema, objetos atómicos generan átomos, conteniendo cada átomo un fragmento especificado de datos o metadatos, mediante lo cual un conjunto de todas las instancias de átomos define de manera colectiva todos los metadatos y los datos en la base de datos. Cada nodo incluye adicionalmente un control de comunicaciones para establecer una trayectoria de comunicaciones con cada uno de los demás nodos en el sistema, un método que responde a una orden de sistema procedente de la interfaz para solicitar de un nodo seleccionado una copia de un átomo que es relevante para la consulta pero que no está presente en ese nodo, un método en respuesta a una petición para que un átomo de otro nodo reproduzca el átomo solicitado para la transferencia al nodo solicitante, mediante lo cual sólo es necesario que los átomos que se requiere que completen una consulta estén ubicados en cualquier nodo de transacción en cualquier tiempo dado, y un método que responde a un cambio en un átomo en ese nodo para reproducir ese átomo para la transferencia a cada uno de los demás nodos en el sistema en el que ese átomo es residente. El almacenamiento persistente contiene una colección de átomos que contiene de manera colectiva todos los datos y los metadatos en la base de datos.

Según otra realización, se proporciona un sistema de gestión de bases de datos que permite que los usuarios interaccionen con una base de datos que se compone de datos y metadatos, dicho sistema incluye al menos un nodo de transacción que proporciona a los usuarios acceso a la base de datos y al menos un nodo de archivado que mantiene un archivo de toda la base de datos. Cada nodo de transacción incluye un motor de petición de base de datos que proporciona una interfaz entre órdenes de consulta de entrada y salida de alto nivel a nivel de usuario y órdenes de entrada y salida a nivel del sistema que controlan una secuencia de operaciones para interaccionar con la base de datos. En respuesta a determinadas órdenes a nivel del sistema, objetos atómicos generan átomos. Cada átomo contiene un fragmento especificado de datos o metadatos, mediante lo cual un conjunto de todas las instancias de átomos define de manera colectiva todos los metadatos y los datos en la base de datos. Una red de sistema de bases de datos interconecta todos los nodos. Un control de comunicaciones en cada uno de los nodos para establecer una trayectoria de comunicaciones con cada uno de los demás nodos en el sistema, un método en cada nodo de transacción responde a una orden de sistema procedente del motor de petición de base de datos para solicitar una copia de un átomo que es relevante para la orden de consulta, pero que no está presente en ese nodo. Otro método en cada nodo responde a una petición para que un átomo de uno de los demás nodos reproduzca el átomo solicitado para la transferencia al nodo solicitante, mediante lo cual sólo es necesario que los átomos que se requiere que completen una orden de consulta estén ubicados en algún nodo de transacción en cualquier tiempo dado. Otro método en cada nodo de transacción responde a un cambio en un átomo en ese nodo para reproducir ese cambio en cualquier otro nodo en el sistema que contiene una copia de ese átomo.

Según aún otra realización un sistema de gestión de bases de datos para una base de datos lógica que se compone de registros de datos organizados en tablas al que va a accederse desde múltiples nodos de transacción que procesan transacciones relacionadas con la base de datos lógica en el que la base de datos se redistribuye en fragmentos en el que cada fragmento almacena una parte de los metadatos y/o los datos que pertenecen a la base de datos lógica para la transferencia dentro del sistema de gestión de bases de datos como mensajes serializados y para el almacenamiento como un mensaje deserializado. El sistema incluye al menos un nodo de archivado que almacena todos los fragmentos de forma deserializada en almacenamiento permanente para constituir de ese modo un único almacén para toda la base de datos. Cada nodo de transacción responde a consultas de los usuarios estableciendo una secuencia de órdenes de bajo nivel para identificar fragmentos que son relevantes para la consulta y responde a las órdenes de bajo nivel obteniendo sólo aquellas copias de fragmentos existentes que son relevantes para la consulta que está procesándose en los mismos mediante, lo cual un fragmento dado puede existir en algún otro nodo o sólo en un nodo de archivado. Cada nodo de transacción reproduce cualquier fragmento cambiado en el al menos un nodo de archivado y cada nodo de transacción en el que reside una copia de ese fragmento, mediante lo cual se realizan cambios en fragmentos en otros nodos en una base de igual a igual y mediante lo cual cualquier nodo de transacción sólo contiene aquellos fragmentos que pertenecen a las consultas realizadas por los usuarios que acceden a la base de datos por medio de ese nodo de transacción.

Breve descripción de los dibujos

Las reivindicaciones adjuntas señalan en particular y reivindican claramente el contenido de esta invención. Los diversos objetos, ventajas y características novedosas de esta invención resultarán más completamente evidentes a partir de la lectura de la siguiente descripción detallada junto con los dibujos adjuntos en los que números de referencia similares se refieren a partes similares, y en los que:

la figura 1 es un diagrama de forma esquemática de una realización de un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido que incorpora esta invención con nodos de transacción y de archivado interconectados;

la figura 2 representa la organización de un nodo de transacción;

la figura 3 representa la organización de un nodo de archivado;

las figuras 4A y 4B representan la organización lógica de objetos de “átomo” generados por las clases de átomo mostradas en las figuras 2 y 3 que son útiles para implementar esta invención y que pueden aparecer en cualquier tiempo dado en un nodo de transacción;

la figura 5 representa la información en un átomo de catálogo maestro;

la figura 6 representa la información en un átomo gestor de transacción;

la figura 7 representa la información en un átomo de base de datos;

la figura 8 representa la información en un átomo de esquema;

la figura 9 representa la información en un átomo de tabla;

la figura 10 representa la información en un átomo de catálogo de tablas;

la figura 11 representa la información en un átomo de índice;

la figura 12 representa la información en un átomo de estados de registro;

la figura 13 representa la información en un átomo de datos;

la figura 14 representa la información en un átomo de estados de Blob;

la figura 15 representa la información en un átomo de Blob;

la figura 16 representa la sintaxis de un mensaje asíncrono a modo de ejemplo que se transfiere entre los nodos de transacción y archivado del sistema de bases de datos de la figura 1;

la figura 17 representa diversos tipos de mensajes mediante los que se transfiere información entre los nodos de transacción y archivado del sistema de bases de datos de la figura 1;

la figura 18 es un diagrama de flujo útil para comprender el método mediante el que un nodo se une al sistema de bases de datos de la figura 1;

la figura 19 representa la información en un objeto de nodo;

la figura 20 es un diagrama de flujo útil para comprender el método mediante el que un nodo crea un átomo según esta invención;

la figura 21 es un diagrama de flujo útil para comprender el método mediante el que se asignan identificaciones únicas de átomo durante el método de la figura 20;

la figura 22 es un diagrama de flujo útil para comprender el método mediante el que un nodo obtiene una copia de un átomo de otro nodo; y

la figura 23 es un diagrama de flujo útil para comprender un método mediante el que esta invención verifica una transacción.

Mejor modo para llevar a cabo la invención

La figura 1 representa una realización de un sistema 30 de bases de datos elástico, variable a escala, bajo demanda, distribuido con una pluralidad de nodos de procesamiento de datos que incorpora esta invención. Los

nodos N1 a N6 son “nodos de transacción” que proporcionan a las aplicaciones de usuario acceso a la base de datos; los nodos A1 y A2, “nodos de archivado” que funcionan para mantener un archivo de disco de toda la base de datos en cada nodo de archivado. Mientras que un nodo de archivado almacena normalmente toda la base de datos, un único nodo de transacción contiene sólo aquella parte de la base de datos que determina que es necesaria para soportar las transacciones que estén realizándose en ese nodo en ese tiempo.

Cada nodo en la figura 1 puede comunicarse directamente con cada uno de los demás nodos en el sistema 30 a través de una red 31 de sistema de bases de datos. Por ejemplo, el nodo N1 puede establecer una trayectoria de comunicaciones con cada uno de los nodos N2 a N6, A1 y A2. Las comunicaciones entre dos nodos cualesquiera son por medio de mensajes serializados. En una realización preferida, la mensajería se realiza de manera asíncrona para maximizar el ancho de banda usado por el sistema para realizar de ese modo diversas operaciones de manera oportuna y rápida. Normalmente, la red 31 de sistema de bases de datos funcionará con una combinación de trayectorias de alto ancho de banda y baja latencia (por ejemplo, una red Ethernet) y trayectorias de alto ancho de banda y alta latencia (por ejemplo, una red WAN). Cada nodo tiene la capacidad de restringir el uso de una trayectoria de baja latencia a las comunicaciones de tiempo crítico (por ejemplo, capturar un átomo). La trayectoria de alta latencia puede usarse para comunicaciones no críticas (por ejemplo, una petición para actualizar la información de una tabla). Además y preferiblemente, la red de procesamiento de datos de esta invención incorpora un protocolo de mensajería, tal como el Protocolo de Control de Transmisión (TCP), y garantiza que cada nodo procese los mensajes en la misma secuencia en la que otros nodos se los enviaron.

La figura 2 representa un nodo 32 de transacción representativo que enlaza con la red 31 de sistema de bases de datos y diversos usuarios 33 finales. El nodo 32 de transacción incluye un sistema 34 de procesamiento central (CP, *central processing*) que se comunica con la red 31 de sistema de bases de datos a través de una interfaz 35 de red y con los diversos usuarios a través de la interfaz 37 de red de usuario. El sistema 34 de procesamiento central también interacciona con la memoria 38 RAM que contiene una copia del programa de gestión de bases de datos que implementa una realización preferida de esta invención. Este programa funciona para proporcionar una interfaz 40 remota, un motor 41 de petición de base de datos y un conjunto 42 de clases u objetos.

El motor 41 de petición de base de datos sólo existe en los nodos de transacción y es la interfaz entre las órdenes de entrada y salida de alto nivel a nivel de usuario y las órdenes de entrada y salida a nivel del sistema a nivel del sistema. En términos generales, su motor de petición de base de datos redistribuye, compila y optimiza las consultas de usuario, tales como las consultas SQL, en órdenes que se interpretan por las diversas clases u objetos en el conjunto 42.

Con propósitos de explicación de esta invención, el conjunto 42 de clases/objetos se divide en un subconjunto 43 de “clases atómicas”, un subconjunto 44 de “clases de mensaje” y un subconjunto 45 de “clases de elemento de ayuda”. Detalles adicionales de estas clases se describen más adelante.

Tal como resultará evidente y según esta invención, en cualquier tiempo dado, un nodo de transacción sólo contiene aquellas partes de la base de datos total que son relevantes para las aplicaciones de usuario activas. Además, las diversas características de esta invención permiten que todas las partes de la base de datos en uso residan en la memoria 38 de acceso aleatorio. No hay necesidad de proporcionar almacenamiento complementario, tal como almacenamiento en disco, en un nodo de transacción durante el funcionamiento de este sistema.

Haciendo referencia a la figura 3, cada nodo 50 de archivado, tal como el nodo A1 o A2 de archivado en la figura 1, también se conecta a la red 31 de sistema de bases de datos. Sin embargo, en lugar de los usuarios 33 finales asociados con un nodo 32 de transacción en la figura 2, un nodo de archivado se conecta sólo al almacenamiento 51 persistente, normalmente un sistema de almacenamiento basado en disco o un almacén de valores clave. El nodo 50 de archivado incluye un sistema 54 de procesamiento central que se comunica con el almacenamiento 51 persistente a través de un canal 52 de I/O y con la red 31 de sistema de bases de datos a través de una interfaz 55 de red. El sistema 54 de procesamiento central también interacciona con la memoria 57 RAM que contiene un conjunto de 62 clases u objetos. De manera similar al nodo 32 de transacción en la figura 2, el conjunto de 62 clases u objetos en la figura 3 incluye un conjunto 63 de “clases atómicas”, un conjunto 64 de “clases de mensajes” y un conjunto 65 de “clases de elemento de ayuda”.

Una realización preferida de esta invención usa programación orientada a objetos (POO) en la que, tal como se conoce en la técnica, clases y subclases tal como se muestra en las figuras 2 y 3 definen métodos, estructuras de datos y procesos mediante los cuales puede generarse una “instancia” u objeto de esa clase o subclase. Puede generarse una “instancia” usando “herencia” y/o “polimorfismo”. Resultará evidente para los expertos en la técnica que son posibles implementaciones que no usan programación orientada a objetos o variaciones en la realización dada a conocer específicamente.

Esta invención se describirá ahora en varias fases. Una sección de “átomos” define la jerarquía y la función de los objetos producidos por las clases 43 y 63 de átomo en las figuras 2 y 3, respectivamente. Una sección de “Mensajes” describe un conjunto de mensajes que proporcionan comunicaciones entre los nodos de transacción y de archivado tal como podrían producirse por las clases 44 y 64 de mensaje en las figuras 2 y 3, respectivamente.

Una sección de “Métodos” describe las operaciones básicas con respecto a la gestión de bases de datos. Una sección de “Ejemplo” describe la interacción del átomo, el mensaje y los métodos mediante los que se logran los objetivos de esta invención en respuesta a una consulta de base de datos específica al motor 41 de petición de base de datos.

Átomos

Tal como se indicó anteriormente, cada una de las clases 43 de átomo en la figura 2 y 63 en la figura 3 producen “átomos”. Más específicamente, las clases de átomo definen uno o más “tipos de átomo” u “objetos de átomo”. Cada “tipo de átomo” u “objeto de átomo” produce una “instancia” de sí mismo, es decir, un “átomo”. Tal como resultará evidente con una comprensión más detallada del propósito de cada objeto atómico específico, cada “átomo” contiene un fragmento específico de información de la base de datos. Algunos átomos contienen una parte de los metadatos de la base de datos; otros contienen registros de datos; otros sirven como catálogos que crean y realizan un seguimiento de otros tipos de átomo. Algunos “tipos de átomo” sólo pueden crear instancias de un átomo que se reproduce en todos los nodos. Otros “tipos de átomo” pueden crear instancias de múltiples átomos que se reproducen en otros nodos según sea necesario.

Los átomos tienen ciertas características. En un nodo de transacción, un átomo existe sólo en la memoria no persistente y en forma de mensaje deserializado que ha rellenado un tipo de átomo particular para proporcionar un formato residente en memoria eficiente para el átomo. Cada átomo tiene medios para codificar su contenido en un mensaje serializado y medios para decodificar un mensaje serializado para recuperar el contenido del átomo. Tales mensajes serializados se usan en conexión con una serie de operaciones tal como se describirá más adelante.

Cada mensaje serializado transmitido desde un nodo para reproducir un átomo incluye el contenido de ese átomo con una identificación de nodo adjunta y el número de secuencia de verificación de transacción más reciente para ese nodo. Cuando un nodo de archivado recibe ese mensaje serializado, deserializa el mensaje, elimina la lista de nodos y verifica el número de secuencia antes de colocar el contenido restante del mensaje (es decir, un átomo) en un almacenamiento persistente.

Se aplican varias reglas a los átomos según esta invención. Las razones e implicaciones de esas reglas resultarán más evidentes. En primer lugar, cada átomo debe tener una identificación única para proporcionar una identificación fiable de ese átomo en cualquier parte de la red 30 de procesamiento de la base de datos en la figura 1. En segundo lugar, cualquier átomo debe existir en dos nodos simultáneamente para mantener la redundancia, excepto que puede existir un único átomo después de que se haya creado y antes de que un nodo de archivo haya solicitado una copia. En tercer lugar, un nodo de transacción cargará un átomo sólo bajo demanda. En cuarto lugar, cada vez que se realiza un cambio en un átomo en un nodo, ese nodo debe reproducir ese átomo cambiado en una base “de igual a igual”; es decir, en todos los nodos de archivado y sólo en los nodos de transacción que contienen ese mismo átomo.

Un proceso de “recopilación de elementos inválidos”, descrito con mayor detalle más adelante, puede suceder en los nodos de archivado y de transacción. El proceso elimina los átomos inactivos de los nodos de transacción y de archivado. Como resultado, un nodo de transacción puede almacenar aquellos átomos que son entonces relevantes actualmente para las aplicaciones de usuario en la memoria de acceso aleatorio en ese nodo. Por tanto, el motor 41 de petición de base de datos “ve” toda la base de datos como local y no sabe que está funcionando en un entorno multinodal y sin una copia completa de la base de datos en su nodo. Los nodos de archivado tienen la opción de purgar el contenido de un átomo después de que se haya serializado en el disco, reduciendo de ese modo el tamaño de la memoria requerida para el almacenamiento. Si un nodo de archivado recibe un mensaje de reproducción para tal átomo, el nodo de archivado debe capturar el contenido del almacenamiento en disco antes de aplicar la información del átomo que está reproduciéndose.

Con estos antecedentes generales, cada tipo de átomo se describirá ahora a un nivel “lógico” o funcional. Esta información, junto con comentarios adicionales sobre el funcionamiento de esta invención, permitirá que un experto habitual en la técnica produzca y use esta invención en cualquiera de una variedad de implementaciones, incluidas implementaciones basadas en programación orientada a objetos.

Las figuras 4A y 4B representan el motor 41 de base de datos y el conjunto típico de átomos que podrían residir en un nodo 32 de transacción en cualquier tiempo dado. En este ejemplo, el nodo de transacción aloja un átomo 70 de catálogo maestro, un átomo 71 gestor de transacción, un átomo 72 de base de datos, un átomo 73 de esquema, un átomo 74 Y de tabla un átomo 75 de catálogo de tablas. Sólo hay un átomo 70 de catálogo maestro, un átomo 71 gestor de transacción y un átomo 72 de base de datos por base de datos. El átomo 71 gestor de transacción, el átomo 72 de base de datos y el átomo 73 de esquema se crean cuando un motor 41 de petición de base de datos crea una nueva base de datos.

Haciendo referencia a la figura 4A, un átomo 70 de catálogo maestro realiza un seguimiento del estatus de los nodos de transacción y de archivado en el sistema 30 de bases de datos de la figura 1. También puede considerarse un índice activo que crea y monitoriza el átomo 71 gestor de transacción, el átomo 72 de base de datos, cada átomo 73

de esquema, cada conjunto correspondiente de átomos 74 Y de tabla átomos 75 de catálogo de tablas, y el gestor 82 de ID de secuencia.

El átomo 75 de catálogo de tablas actúa como índice activo y crea y monitoriza los átomos 76 de índice, los átomos 77 de estados de registro, los átomos 78 de datos, los átomos 80 de los estados Blob y los átomos 81 de Blob asociados con una única tabla. Es decir, hay un átomo 75 de catálogo de tablas para cada tabla.

La figura 4B es útil para comprender la interacción y la gestión de diferentes tipos de átomo. En este contexto, ni el átomo 70 de catálogo maestro ni el átomo 75 de catálogo de tablas realizan funciones de gestión. Con respecto a los átomos restantes, el átomo 70 de base de datos gestiona cada átomo 73 de esquema. Cada átomo 73 de esquema gestiona cada átomo 74 de tabla relacionado y los átomos 82 gestores de ID de secuencia. Cada átomo 74 de tabla gestiona su átomo 75 de catálogo de tablas correspondiente, átomos 76 de índice, átomos 77 de estados de registro, átomos 78 de datos, átomos 80 de estados de Blob y átomos 87 de Blob.

Todavía haciendo referencia a la figura 4B, el motor 41 de petición de base de datos se comunica con el átomo 70 de catálogo maestro, el átomo 71 gestor de transacción, el átomo 72 de base de datos, cada átomo 73 de esquema, cada átomo 74 Y de tabla los gestores 82 de ID de secuencia. El motor 41 de petición de base de datos actúa como compilador para un lenguaje de alto nivel tal como SQL. Como compilador, redistribuye, compila y optimiza consultas y obtiene metadatos y datos de los átomos para la formación de los diversos fragmentos de información de la base de datos.

Cada átomo tiene ciertos elementos comunes y otros elementos que son específicos de su tipo. Haciendo referencia a la figura 5, un átomo 70 de catálogo maestro incluye elementos 70A a 70I comunes. El elemento 70A es una identificación única para el átomo. Como sólo hay una instancia de un átomo de catálogo maestro que se reproduce en todos los nodos, a la ID 70A de átomo de catálogo maestro se le facilita un número fijo, normalmente “1”. Como regla general, los punteros 70B y 70C identifican el átomo de catálogo maestro y el átomo de catálogo de creación, respectivamente. Para el átomo de catálogo maestro, ambos punteros identifican el propio átomo de catálogo maestro.

Cada átomo debe tener un director. El director realiza funciones tal como se describe más adelante. El elemento 70D es un puntero para aquel nodo en el que reside el director de ese átomo.

Cada vez que se cambia una copia de un átomo en cualquier nodo de transacción, recibe un nuevo número de cambio. El elemento 70E registra ese número de cambio.

Siempre que un nodo solicita un átomo de otro nodo, hay un intervalo durante el cual el nodo solicitante no será conocido por otros nodos de transacción. El elemento 70F es una lista de todos los nodos para los que el nodo de suministro debe transmitir mensajes al nodo solicitante desde todos los demás nodos que contienen ese átomo hasta que se complete la petición.

Las operaciones del sistema de bases de datos también se dividen en ciclos. Un elemento 70G de referencia de ciclo proporciona el número de ciclo del último acceso al átomo. El elemento 70H es una lista de todos los nodos activos que contienen el átomo. El elemento 70I incluye varios indicadores de estatus.

Todavía haciendo referencia a la figura 5, una entrada 70J de IDS de nodo global contiene un gestor de ID para asignar un identificador único para cada nodo activo en el sistema. Tal como se sabe, tales identificadores son largas cadenas de caracteres. Una entrada 70K de IDS de nodo local incluye un rango de números hasta el número total de nodos que pueden unirse al sistema. Juntas, estas entradas proporcionan la correspondencia entre los dos tipos de identificación. El uso de IDS de nodo local aumenta la eficiencia.

Cuando un nodo de transacción se une al sistema 30 de bases de datos en la figura 1, un gestor 70L de conexión efectúa ese proceso. Más adelante se describe una implementación específica para permitir que el nodo de transacción se una al sistema de bases de datos. El nodo de unión usa una entrada 70M de estatus de nodo pendiente para indicar que no tendrá una dirección global para un nodo de respuesta hasta que reciba comunicaciones adicionales desde ese nodo de respuesta. Una entrada 70N de UUI de base de datos contiene la identificación única universal para la base de datos.

Las entradas en 70P son importantes porque vinculan todos los átomos para los que el átomo 70 de catálogo maestro actúa como índice activo. Tal como se indicó anteriormente, estos incluyen el átomo 72 de base de datos y cada uno de los átomos 73 de esquema, los átomos 74 Y de tabla los átomos 75 de catálogo de tablas.

Una entrada 70Q de contraseña es representativa de un medio para autenticar una conexión en la base de datos. Las entradas 70R y 70S de versión real y de software permiten que el sistema funcione con compatibilidad con versiones anteriores cuando se instala una versión más reciente del software. La entrada 70R de software real identifica la versión de software en uso entonces; la entrada 70S de versión de software, el número correspondiente a la versión instalada más reciente. Esto permite que los nodos individuales se actualicen a versiones más nuevas

sin requerir que se actualicen otros nodos y sin cerrar la base de datos para el acceso por todos los nodos.

Todavía haciendo referencia a la figura 5, el átomo 70 de catálogo maestro también incluye un puntero 70T para el átomo 71 gestor de transacción, un puntero 70U para un objeto de configuración, un puntero 70V para un subproceso de recopilación de elementos inválidos y un puntero 70W para un subproceso de *ping*. El subproceso de *ping* funciona periódica e independientemente de otras operaciones en un nodo. Se "hace *ping*" entre cada uno de los demás nodos para proporcionar información que pueda usarse en la determinación de la eficiencia de las comunicaciones de una trayectoria correspondiente. Por ejemplo, si el nodo N1 en la figura 1 tiene la opción de comunicarse con el nodo N2 o N5, el nodo N1 podría usar la información de *ping* en la selección de las trayectorias de comunicación más eficientes hasta los nodos N2 y N5 para esa comunicación. Otros procesos de selección también pueden sustituirse o añadirse.

Haciendo referencia a la figura 6, hay un átomo 71 gestor de transacción para cada base de datos y se crea durante el mismo proceso que crea el átomo 70 de catálogo maestro. El átomo 71 gestor de transacción crea, realiza un seguimiento y finaliza las transacciones de la base de datos en respuesta a las órdenes de la base de datos procedentes del motor 41 de petición de base de datos. Un átomo 71 gestor de transacción incluye elementos 71A-71I que corresponden a elementos similares en el átomo de catálogo maestro. Sin embargo, el elemento 71A es la identificación del átomo 71 gestor de transacción. Los elementos 71B y 71C apuntan ambos al átomo 70 de catálogo maestro.

Un gestor 71J de ID proporciona identificaciones de secuencia de transacción únicas y mantiene una lista 71K de transacciones activas, una lista 71L de transacciones verificadas y una lista 71M de transacciones fallidas. El elemento 71N almacena información de secuencia de verificación. El gestor 71J de ID asigna una ID de transacción al inicio de cada transacción. Cada ID de transacción es única, pero no necesariamente secuencial. Un átomo gestor de transacción local asigna un número de secuencia de verificación al elemento 71N cuando se verifica una transacción. Los números de secuencia son secuenciales y cada uno es específico del nodo que solicitó la transacción. Un contador 71P de eventos de transición y transacción identifica eventos discretos que suceden durante cada transacción, tales como el inicio de la transacción y la verificación satisfactoria de la transacción. Tales contadores son útiles cuando se solapan varias transacciones que implican la misma información.

Haciendo referencia a la figura 7, el átomo 72 de base de datos se crea al mismo tiempo que se crean el átomo 70 de catálogo maestro y el átomo 71 gestor de transacción. El átomo 72 de base de datos identifica cada uno de los átomos 73 de esquema. El átomo 72 de base de datos puede estar implicado en un proceso de autenticación cuando un nuevo usuario intenta unirse a la base de datos. También puede incluir otros datos referentes a los niveles de autorización.

Básicamente, el átomo 72 de base de datos incluye elementos 72A-72I correspondientes a elementos similares en la figura 5. El elemento 72A es la identificación del átomo de base de datos. Cada uno de los punteros 72B y 72C identifica el átomo 70 de catálogo maestro. Un registro 72J de nombre de esquema e ID de esquema relaciona los nombres de esquema con las identificaciones de átomo de esquema.

Haciendo referencia a la figura 8, un átomo 73 de esquema crea y realiza un seguimiento de átomos de tabla para ese esquema. Un átomo 72 de base de datos puede gestionar múltiples átomos de esquema y cada átomo de esquema puede interaccionar con múltiples átomos de la tabla. El átomo 73 de esquema incluye elementos 73A-73I correspondientes a los elementos 70A-70I en la figura 5. El elemento 73A es la identificación 73A única de átomo de esquema y los elementos 73B y 73C son punteros para el átomo 70 de catálogo maestro. Una tabla tiene un nombre único dentro de un esquema. Un registro 73J de nombre de tabla e ID de átomo de tabla proporciona las correspondencias entre cada nombre de tabla y el átomo de tabla correspondiente. Cada secuencia de esquema tiene un nombre. Un registro 73K de nombre de secuencia y gestor de ID de secuencia proporciona la relación entre esos nombres y los gestores de ID de secuencia correspondientes asociados con cada átomo de esquema, tales como los gestores 82 de ID de secuencia en las figuras 4A y 4B.

La figura 9 proporciona una vista lógica de un átomo 74 de tabla que incorpora metadatos relacionados con campos, formatos, índices y tipos y que gestiona cada uno de los átomos 76 de índice, átomos 77 de estados de registro y átomos 80 de estados de Blob para esa tabla. También crea y realiza un seguimiento de datos dentro de una tabla. El átomo 74 de tabla incluye elementos 74A-74I que corresponden a los elementos 70A-70I en la figura 5. El elemento 74A incluye la identificación única de átomo de tabla, los elementos 74B y 74C apuntan ambos al átomo de catálogo maestro. El puntero 74J identifica el átomo de catálogo de tablas correspondiente. El elemento 74K contiene una lista de todos los campos para la tabla.

Cada átomo de tabla tiene varios gestores de ID. Un puntero 74L apunta a un gestor de ID que proporciona a cada campo una identificación única. Los punteros 74M, 74N, 74P y 74Q identifican gestores de ID independientes para asignar identificaciones a átomos de índice, átomos de datos, átomos de Blob y subtipos, respectivamente. El elemento 74R es una lista de los subtipos existentes. Las matrices 74S y 74T proporcionan las ubicaciones de los átomos de estados de registro y los átomos de estados de Blob, respectivamente.

Ahora haciendo referencia a la figura 10, hay un átomo de catálogo de tablas para cada átomo de tabla. Cada átomo 75 de catálogo de tablas se crea cuando se crea un átomo de tabla. A su vez, un átomo de catálogo de tablas crea y realiza un seguimiento de átomos específicos de una tabla, incluidos los átomos de índice, de estados de registros, de datos, de estados de Blob y de átomos de Blob. Cada átomo 75 de catálogo de tablas incluye elementos 75A-75I que corresponden a los elementos 70A-70I en la figura 5. El elemento 75A es una identificación única de átomo de catálogo de tablas asignada por el átomo de catálogo maestro. Ambos elementos 75B y 75C apuntan al átomo 70 de catálogo maestro. Un gestor 75J de ID proporciona identificaciones únicas de átomo para cada uno de los átomos de índice, de estados de registro, de datos, de estados de Blob y de Blob. Una lista 75K identifica todos los átomos asociados con un átomo de tabla correspondiente. Los punteros en el elemento 75L identifican la ubicación de cada átomo en el nodo local asociado con el átomo de tabla correspondiente. Una serie de listas 75M identifican, para cada átomo, una lista de los nodos con reproducciones de ese átomo. Los mapas 75N de bits proporcionan un medio conveniente para identificar otros objetos y directorios cuando el átomo está en un nodo de archivado.

Haciendo referencia a la figura 11, hay un átomo 76 de índice para cada índice en una tabla, y puede haber múltiples átomos de índice por tabla. Cada átomo de índice incluye elementos 76A-76I que corresponden a los elementos 70A-70I en la figura 5, respectivamente. El elemento 76A es la identificación única de átomo de índice asignada por el átomo de catálogo de tablas correspondiente. Los punteros 76B y 76C identifican el átomo de catálogo maestro y el átomo de catálogo de tablas, respectivamente. El elemento 76J contiene un árbol binario de nodos de índice para proporcionar una función de indexación convencional. El elemento 76K contiene el nivel de índice. Tales operaciones y estructuras de índice las conocen los expertos en la técnica.

Haciendo referencia a la figura 12, un átomo 77 de estados de registro gestiona las versiones de registro y el estado para un rango fijo de números de registro dentro de una única tabla. Por tanto, un átomo de tabla dado puede gestionar múltiples átomos de estados de registro. Cada átomo de estados de registro incluye elementos 77A-77I que corresponden a los elementos 70A-70I en la figura 5. El elemento 77A incluye la ID de átomo de estados de registro que asigna el átomo de catálogo de tablas de creación. Los punteros 77B y 77C identifican los átomos de catálogo maestro y de catálogo de tablas, respectivamente. El elemento 77J es una matriz para ubicar todos los átomos de datos gestionados por el átomo 77 de estados de registro. El elemento 77K contiene el número de registro para un “registro base”. Es decir, cada átomo de datos almacena múltiples registros. El elemento 77 es un puntero para el átomo de tabla correspondiente.

En las aplicaciones de bases de datos a las que se refiere esta invención, múltiples usuarios pueden producir múltiples versiones del mismo registro. Una realización preferida de esta invención usa el control de concurrencia multiversión (MVCC, por sus siglas en inglés) para garantizar que una transacción nunca tenga que esperar a la base de datos permitiendo que existan simultáneamente varias versiones de un registro u otro objeto en la base de datos. Por consiguiente, cada átomo 77 de estados de registro incluye metadatos sobre cada versión de registro. La entrada 77M es un mapa de bits para identificar la ubicación de cada registro con versión que es útil en la recopilación de elementos inválidos.

El átomo 77 de estados de registro incluye, para cada versión 77N de un registro, una ID 77P de transacción para identificar la transacción que generó la versión. Una entrada 77Q de versión de formato identifica el número de versión para el subtipo de tabla que existía cuando se insertó el registro. Este formato identifica el orden físico del registro e identifica el subtipo al que pertenece el registro del programa de base de datos que estaba en uso en el momento en que se creó la versión del registro. El elemento 77R incluye un número de secuencia de versión de registro; el elemento 77S, la ubicación de la siguiente versión más antigua, o previa, del registro. Un índice 77T para la matriz 77J de átomos de datos y la identificación 77K de registro base proporcionan conjuntamente la dirección para la ranura 77U real en el átomo de datos con la versión de registro.

La figura 13 representa un átomo 78 de datos con elementos 78A-78I que corresponden a los elementos 70A-70I en la figura 5. En el átomo 78 de datos, el elemento 78A es la identificación 78A de átomo de datos asignada por el átomo de catálogo de tablas. Los elementos 78B y 78C son punteros para el átomo de catálogo maestro y para el átomo de catálogo de tablas correspondiente, respectivamente. Un gestor 78J de ID asigna una identificación de ranura de registro para cada registro en el átomo 78 de datos. El elemento 78K identifica, para cada registro en el átomo 78 de datos, la dirección y la longitud de ese registro. El elemento 78C representa registros de datos y versiones de los mismos.

Ahora haciendo referencia a la figura 14, las bases de datos también almacenan “registros de Blob”. Un “registro de Blob” es normalmente una colección de datos binarios almacenados como una única entidad en la base de datos. Los registros de Blob no existen en las versiones. Un átomo de estado de Blob incluye elementos 80A-80I que corresponden a los elementos 70A-70I en la figura 5. El elemento 80A tiene la identificación única de átomo del átomo de estados de Blob. Los elementos 80B y 80C son punteros para los átomos de catálogo maestro y de catálogo de tablas, respectivamente. La lista 80J identifica todos los átomos de Blob gestionados por un único átomo 80 de estados de Blob. La entrada 80K proporciona la identificación del registro de Blob base. El elemento 80L apunta al átomo de tabla correspondiente. Para cada registro de Blob, el átomo de estados de Blob incluye un índice 80M para el átomo de Blobs. El elemento 80N identifica la ranura en un átomo de Blob para el registro de Blob.

La figura 15 representa un átomo 81 de Blob con elementos 81A-81I que corresponden a los elementos 70A-70I en la figura 5, respectivamente. El elemento 81A es la identificación de átomo asignada por el átomo de catálogo de tablas. Los elementos 81B y 81C son punteros para el átomo de catálogo maestro y para el átomo de catálogo de tablas correspondiente, respectivamente. Un gestor 81J de ID asigna una identificación de ranura de Blob a cada Blob en el átomo 81 de Blob. El elemento 81K identifica, para cada Blob en el átomo 78 de Blob, su dirección y longitud. El Elemento 81L representa todos los registros de Blobs asignados al átomo de Blob.

En resumen, cada átomo tiene una relación sólo con un fragmento de la base de datos. Por ejemplo, el átomo 72 de base de datos contiene metadatos que identifican el esquema para la base de datos. Cada átomo 73 de esquema contiene metadatos que identifican todas las tablas asociadas con ese esquema. Para cada tabla, un átomo 74 Y de tabla un átomo 75 de catálogo de tablas correspondiente proporcionan metadatos sobre la tabla que incluyen información tal como la identificación de campos y sus propiedades. Los átomos de estados de registro incluyen metadatos sobre un grupo de registros. Los átomos de datos incluyen información sobre cada registro de datos con punteros para las ranuras que contienen estos registros y diversas versiones. Los átomos de estados de Blob y de Blob contienen información similar sobre los registros de Blob.

Mensajes

Tal como se indicó anteriormente, las comunicaciones entre dos nodos se realizan a través de mensajes serializados que se transmiten de manera asíncrona usando el TCP u otro protocolo con controles para mantener las secuencias de mensajes. La figura 16 representa la sintaxis básica de un mensaje 90 típico que incluye una cabecera 91 de longitud variable y un cuerpo 92 de longitud variable. La cabecera 91 incluye un código 93 de identificador de mensaje que especifica el mensaje y su función. Como esta invención prevé un escenario en el que diferentes nodos pueden funcionar con diferentes versiones de software, la cabecera 91 también incluye una identificación 94 de la versión de software que creó el mensaje. Los elementos restantes en la cabecera incluyen una identificación 95 local del remitente (es decir, del átomo de catálogo maestro en la figura 5) e información 96 para el destino del mensaje, concretamente un átomo de catálogo (por ejemplo, una ID 75A de catálogo de tablas en la figura 10) y una identificación 97 de átomo (por ejemplo, una ID 77A de estados de registro en la figura 12). A partir de esta información, el nodo receptor puede deserializar, decodificar y procesar el mensaje.

La figura 17 representa un conjunto de mensajes que tienen la sintaxis de la figura 16 para una realización específica de esta invención. Cada uno realiza una función específica tal como se describirá ahora.

Tal como se comentó brevemente con anterioridad, cuando va a enviarse un mensaje, hay diferentes trayectorias de comunicaciones hasta diferentes nodos. Por ejemplo, si un nodo, como nodo solicitante, requiere obtener un átomo, las reproducciones de ese átomo pueden ubicarse en múltiples de otros nodos. En esta realización, “hacer *ping*” proporciona información de selección útil para seleccionar el nodo correspondiente de la mejor trayectoria. Hacer *ping*, tal como se sabe, implica determinar el tiempo para que una orden de “*ping*” llegue a su destino y para recibir un mensaje de acuse de recibo. En esta realización de la invención, cada nodo usa periódicamente una clase de elemento de ayuda para enviar un mensaje 110 de *ping* a cada uno de los otros nodos a los que se conecta. Cada nodo receptor usa una clase de elemento de ayuda para devolver un mensaje 111 de acuse de recibo de *ping* que contiene el tiempo de *ping*. Cada nodo acumula esta información sobre el tiempo para enviar y recibir estos mensajes en un objeto de nodo descrito más adelante con respecto a la figura 8. Cuando un nodo se prepara para enviar un mensaje a uno de múltiples nodos, el nodo transmisor analiza factores que incluyen, pero no se limitan a, los datos de *ping* acumulados para seleccionar uno de los nodos como nodo receptor para ese mensaje.

Un siguiente conjunto de mensajes está implicado con la conexión de un nuevo nodo en un nodo inactivo anteriormente de vuelta en el sistema 30 de bases de datos en la figura 1. Cuando tal nodo, por ejemplo, el nodo N2 de transacción, desea conectarse al sistema 30 de bases de datos, inicia un proceso de conexión descrito con detalle más adelante con respecto a la figura 19. Haciendo referencia a la figura 17, una vez que ese proceso identifica un nodo activo para recibir mensajes, el nodo de unión envía un mensaje 112 de conexión al nodo seleccionado. El nodo seleccionado devuelve un mensaje 113 de bienvenida y un mensaje de nuevo nodo 114 a todos los demás nodos conectados en el sistema 30 de bases de datos. Cada uno de los otros nodos conectados transmite su propio mensaje 113 de bienvenida al nodo de unión. Cuando se completa esta secuencia de mensajes, el nodo de unión puede tomar medidas adicionales para obtener diversos átomos.

Los nodos de archivado pueden funcionar en modo activo o en modo de sincronización cuando se sincronizan con otro nodo. Una clase de elemento de ayuda en el nodo de archivado transmite un mensaje 115 de estado de nodo para alertar a todos los demás nodos de cualquier cambio de estatus en ese nodo de archivado.

Un siguiente conjunto de mensajes está implicado cuando un nodo, como nodo solicitante, recupera una copia de un átomo de otro nodo. Por ejemplo, después de que un nodo se une al sistema 30 de bases de datos en la figura 1, solicita generalmente una copia del átomo de catálogo maestro. El proceso se describe con mayor detalle en relación con la explicación de las figuras 20 y 21.

Todavía haciendo referencia a la figura 17, el nodo solicitante emite un mensaje 116 de petición de objeto a un nodo

seleccionado que devuelve generalmente un mensaje 117 de objeto con el átomo solicitado. El nodo seleccionado también envía un mensaje 118 de objeto disponible a todos los demás nodos con ese átomo. Cada nodo que recibe un mensaje 118 de objeto disponible desde el nodo seleccionado devuelve un mensaje 119 de acusado recibo de objeto al nodo seleccionado. El nodo seleccionado envía un nodo 120 de mensaje completo al nodo solicitante después de que el nodo seleccionado recibe todos los mensajes 119 de acusado recibo de objeto.

En algunas situaciones, el nodo seleccionado envía un mensaje 121 de objeto no disponible para anunciar que el nodo seleccionado ha suprimido el átomo solicitado. Un mensaje de objeto de devolución desde el nodo seleccionado indica que el átomo solicitado no se encuentra en el átomo de catálogo maestro ni en uno de los átomos de catálogo de tablas. Esto puede suceder cuando una transacción de actualización está en curso y el nodo seleccionado no responde al mensaje 116 de petición de objeto porque el proceso de recopilación de elementos inválidos ha recopilado ese átomo antes de recibir el mensaje de petición de objeto. Como respuesta, el nodo de transacción solicitante puede seleccionar otro nodo en el sistema de bases de datos con el átomo.

El motor 41 de petición de base de datos en la figura 1 y las figuras 4A y 4B pueden generar periódicamente un mensaje 123 de registrar objeto o un mensaje 124 de cancelar registro de objeto. Estos mensajes están dirigidos a átomos que incluyen un registro, como los átomos de base de datos y de esquema. Se envía un mensaje 125 de suprimir objeto desde un nodo cuando un usuario en ese nodo emite una orden para suprimir algún elemento, tal como una tabla.

Cada vez que un nodo local se actualiza o modifica, su número de cambio se incrementa. Cada mensaje de reproducción contiene el número de cambio local de ese átomo. Cada átomo realiza un seguimiento del número de cambio más reciente para cada nodo. Cuando un nodo de archivado recibe una copia de un átomo modificado, copia los números de cambio para los átomos, borra los números de cambio y luego serializa el átomo en el disco. Si se ha realizado un cambio, el número de cambio no será cero. El nodo de archivado envía un mensaje 126 de objeto escrito con el número de cambio escrito en cada nodo. Cada nodo receptor compara su propio número de cambio con el del mensaje. Luego, el nodo puede actualizar el estatus para el átomo para observar que el átomo se ha archivado y es un candidato potencial para la recopilación de elementos inválidos.

Tal como se indicó anteriormente, cada átomo debe tener una identidad única. Cuando se crea un primer átomo de un tipo de átomo particular (por ejemplo, un nuevo átomo de tabla), el nodo de transacción de creación se designa como director para ese átomo de tabla. Las reglas para regir la “dirección” se comentan más adelante. Cada vez que es necesario que un nodo de transacción cree un nuevo tipo de átomo, envía un mensaje 127 de petición de ID al director si no tiene una identificación disponible. El director devuelve generalmente un mensaje 128 de delegación de ID que incluye un bloque de al menos una identificación única de su atribución de valores de identificación libres. Este proceso se describe con mayor detalle con respecto al proceso de la figura 20.

Haciendo referencia a las figuras 4A y 4B, un átomo 74 de tabla puede enviar cualquiera de varios mensajes. Si el motor 41 de petición de base de datos inicia un proceso para añadir un nuevo campo, el átomo de tabla en ese nodo de transacción correspondiente genera la nueva estructura y envía un mensaje 129 de campo de tabla añadido que reproduce el cambio en todos los demás nodos que incluyen ese átomo de tabla. Si un nodo de transacción actualiza un campo, un subtipo o una característica de campo que cambia el formato de tabla, ese nodo emite un mensaje 130 de formato de tabla. Se genera un mensaje 131 de registros de petición de tabla siempre que es necesario que un nodo cree un nuevo átomo de estados de registro o de estados de Blob. Sólo un director puede crear este átomo; y el director difunde un mensaje 132 de objeto de registros de tabla cuando esto sucede.

Cada vez que un nodo de transacción inserta un nuevo registro en una tabla, produce un mensaje 133 de registro de tabla. Siempre que resulte necesario crear un nuevo índice, tal como cuando se añade un campo indexado a una tabla, el átomo de tabla de creación reproduce el nuevo átomo de tabla. En este tiempo, el índice está configurado para ser un índice de sólo escritura. Una vez que se han completado todos los procesos relacionados implicados, el nodo envía un mensaje 134 de índice de tabla añadido.

Cada vez que un átomo de tabla crea un átomo de estados de Blob para un Blob, el átomo de tabla genera un mensaje 135 de objeto de Blob de tabla. Un mensaje 136 de Blob de tabla indica que se ha creado un nuevo Blob.

En una base de datos que utiliza tipos y subtipos, el motor 41 de petición de base de datos en la figura 1 generará órdenes para hacer que un átomo de tabla asigne una nueva identificación de tipo de tabla. Cuando esto sucede, un mensaje 137 de tipo de tabla se reproduce en todos los nodos con átomos de tabla similares.

Un mensaje 138 de borrar registros de tabla proporciona el número de registro en un átomo de estados de registro específico. Un director genera un mensaje 139 de recopilar elementos inválidos de tabla para un átomo de tabla cuando se determina que un registro dentro de esa tabla contiene una cadena inusualmente larga de versiones anteriores u otros criterios. El resultado es que los átomos “no usados” se “vacían” de datos.

Un átomo 77 de estados de registro en las figuras 4A y 4B también produce varios mensajes especializados. Si resulta necesario actualizar un registro particular, el sistema de gestión de bases de datos de esta invención crea

una nueva versión de ese registro. Volviendo a hacer referencia a la figura 17, el átomo de estados de registro correspondiente genera un mensaje 140 de petición de actualización de registro que se dirige al director para que ese átomo solicite permiso para actualizar ese registro en particular. El director responde generando un mensaje 141 de respuesta de actualización de registros que concede o deniega el permiso para actualizar ese registro. Si el director concede permiso, el átomo de estados de registro solicitante realiza la actualización y envía un mensaje 142 de actualización de registros con la nueva versión del átomo a cada nodo con una copia de ese átomo.

Un átomo de datos almacena hasta cierto número máximo de registros y versiones de los mismos. Un átomo de estados de registro correspondiente monitoriza el tamaño de los átomos de datos que gestiona. Si un átomo de estados de registro determina que un átomo de datos gestionado ha superado ese tamaño, genera un nuevo átomo de datos y reproduce ese nuevo átomo de datos mediante un mensaje 143 de objeto de datos de registros.

Un átomo de estados de registro genera un mensaje 144 de registro de registros para reproducir cualquier nueva versión de registro. Periódicamente, un director de átomo de tabla inicia un proceso mediante el que un átomo de estados de registro identifica versiones de registro que son más antiguas que la transacción activa más antigua. Cuando esto sucede, el átomo de estado de registro del director transmite un mensaje 145 de eliminación de registros que permite que esas versiones de registros más antiguas se supriman durante un proceso de recopilación de elementos inválidos posterior. Si surge una situación en la que resulta necesario revertir una transacción, un átomo de estados de registro genera un mensaje 146 de registro de anulación que actualiza el registro de anulación.

También hay un conjunto de mensajes específicos de índice. Tal como se sabe, un índice tiene un tamaño máximo óptimo; y si el índice supera ese tamaño, el índice debe dividirse. Según esta invención, sólo un director puede dividir un átomo de índice. El director puede realizar esto unilateralmente o en respuesta a un mensaje 147 de petición de división de índice de otra copia de ese átomo. Cuando el director provoca la división, el director genera un mensaje 148 de división de índice que contiene un índice dividido. Después de la división, el director envía un mensaje 149 de eliminación de índice para truncar el átomo de índice original después de la división. Cada vez que se añade un nodo de índice al índice, genera un mensaje de nodo de índice añadido que contiene una clave de índice, identificación de registro y otra información para el nuevo índice. Cuando un índice se ha rellenado por completo y, por tanto, está preparado para su uso durante operaciones de recuperación, se genera un mensaje 151 de índice de tabla preparado. Tal como se indicó anteriormente, añadir un índice de tabla genera un átomo de índice que es un índice de sólo escritura y que no es legible. El mensaje 151 de índice de tabla preparado hace que tal índice de sólo escritura sea legible.

Un átomo de estados de Blob identifica una ranura en un átomo de Blob generando un mensaje 152 de Blob de Blob.

Un mensaje 153 de registro de datos contiene el número de ranura y la longitud del registro. También incluye el registro de datos.

A medida que cambia el estado de una transacción, el átomo 71 gestor de transacción para un nodo de transacción dado genera mensajes 154 de estado de transacción. Estos indican si el estatus actual de la transacción está en un estado activo, un estado de verificación previa, un estado de verificación o un estado de reversión. En determinadas circunstancias, es posible que una petición relacionada con una transacción sea bloqueada por otra transacción en otro nodo. En ese caso, el átomo gestor de transacción recibe un mensaje 155 de bloqueo de transacción. Si una transacción está bloqueada y debe revertirse, el átomo gestor de transacción asociado con el nodo que causa el bloqueo genera un mensaje 156 de interbloqueo de transacción que hace que la transacción se revierta.

El átomo de catálogo maestro o cualquier átomo de catálogo de tablas en un nodo de archivado puede solicitar la última vez que se escribió un átomo. Esto sucede cuando un nodo solicitante envía un mensaje 158 de tiempos de escritura de petición. El receptor de ese mensaje devuelve los tiempos solicitados en un mensaje 159 de tiempos de escritura.

Los mensajes anteriores constituyen un conjunto mediante el que los diversos procedimientos necesarios para mantener un sistema de gestión de bases de datos que incorpora esta invención pueden manejarse adecuadamente. Tal como resultará evidente, cada mensaje tiene un mínimo de tara en una cabecera. Cada mensaje puede ser razonablemente corto. Cuando se usa con TCP u otro protocolo de mensajería, los mensajes deben enviarse secuencialmente y, en cualquier nodo dado, tras su recepción deben manejarse en la misma secuencia en la que se enviaron incluso por diferentes nodos.

Métodos

Ahora será útil comprender mejor esta invención para describir algunos métodos básicos que se refieren a diversos aspectos del funcionamiento de esta invención. Resultarán evidentes variaciones de cada uno para los expertos en la técnica.

La figura 18 es un diagrama de flujo de las operaciones que suceden en diferentes nodos en el sistema 30 de bases

de datos de la figura 1 cuando un nodo se une a la red. A efectos de esta descripción, supóngase que el nodo 5 en la figura 1 va a unirse al sistema de bases de datos; está designado para ser el nodo 170 de transacción de unión. Como primer proceso, el nodo 170 de transacción de unión usa la etapa 171 para establecer una conexión TCP con un nodo seleccionado. Básicamente, el nodo 170 de transacción de unión envía un mensaje a una ubicación fija que identifica una base de datos. Un agente de conexión, que no se muestra pero se conoce en la acción, responde a esta petición denegando o concediendo acceso al sistema de bases de datos. Si el agente de conexión concede acceso, selecciona un nodo, tal como el nodo N1 de transacción, tal como un nodo 172 seleccionado. Luego, el agente de conexión envía un mensaje al nodo 170 de transacción de unión con una designación, como mediante un número de puerto, del nodo 172. La figura 18 también representa como grupo 173 todos los demás nodos de transacción y de archivado activos en el sistema de bases de datos.

Una vez que se establece la conexión en la etapa 171, el nodo 170 de transacción de unión usa la etapa 174 para enviar un mensaje de conectar al nodo 172 seleccionado. El nodo 172 seleccionado responde al mensaje de conectar en la etapa 175 actualizando su átomo de catálogo maestro con su gestor de conexión, asignando una ID de nodo local al nodo 170 de transacción de unión y añadiendo ese nodo a una matriz de nodos locales en su objeto de nodo, se muestra un ejemplo en la figura 19.

La figura 19 representa un objeto de nodo tal como se describe en conexión con la etapa 175 de la figura 18. Contiene un puntero 400A para el conector para el nodo y un puntero 400B para el átomo de catálogo maestro en el nodo y un puntero 400C para el gestor de conexión en el átomo de catálogo maestro. El puntero 400D identifica el subproceso usado para escuchar los mensajes entrantes y el puntero 400E identifica una memoria intermedia de conectores para recibir mensajes. El agente de escucha de mensajes espera un mensaje, determina el tipo de mensaje y luego procesa el mensaje hasta que se completa.

El objeto 400 de nodo, como todos los átomos, incluye una ID 400F de nodo global y una ID 400G de nodo local del nodo al que escucha este nodo. El elemento 400H es un puntero para una cola de mensajes que esperan ser enviados desde el nodo. Los elementos 400I y 400J contienen las identificaciones para el puerto local y los puertos remotos. El elemento 400K contiene el número de versión para el software que funciona en el nodo remoto. Un elemento 400L de tipo de nodo indica si el nodo remoto es un nodo de transacción, un nodo de archivado en proceso de sincronización o un nodo de archivado en línea.

El elemento 400M contiene el nombre para el nodo local; el elemento 400N, el nombre para el nodo remoto. El elemento 400P es un puntero para el mensaje actual que está procesándose. Los elementos 400Q y 400R identifican el tiempo de la última operación de *ping* y el tiempo de *ping*. Tal como se indicó anteriormente, cada nodo genera un número de secuencia de verificación secuencial en respuesta a cada operación de verificación para una transacción iniciada en ese nodo. El elemento 400S contiene este número. El elemento 400T indica si este objeto de nodo es el objeto de nodo para este nodo.

Volviendo a hacer referencia a la figura 18, en la etapa 176, el nodo 172 seleccionado envía un mensaje de bienvenida al nodo 170 de transacción de unión que contiene la ID de nodo global. Luego, el nodo seleccionado difunde un mensaje de nuevo nodo en la etapa 177 al grupo 173 de todos los demás nodos de transacción y de archivado.

Cada nodo en el grupo 173 responde al mensaje de nuevo nodo registrando la ID global y asignando una ID local del nodo de transacción de unión y actualizando una lista local de todos los nodos conectados en sus átomos de catálogo maestro respectivos en la etapa 180. Cada uno usa entonces la etapa 181 para enviar un mensaje de bienvenida al nodo 170 de transacción de unión. Tras completarse este proceso, el nodo 170 de transacción de unión tiene una lista completa de todos los nodos activos, incluido el nodo 172 seleccionado y todos los nodos del grupo 173.

Cuando el nodo 170 de transacción de unión recibe el mensaje de bienvenida procedente del nodo seleccionado en la etapa 182, envía un mensaje de petición de objeto (etapa 183) al nodo 172 seleccionado que solicita una copia del átomo de catálogo maestro. Después de que el nodo 172 seleccionado haya actualizado los diversos elementos de información en su átomo de catálogo maestro, el nodo seleccionado implementa la etapa 184 para serializar su átomo de catálogo maestro en un mensaje de objeto que se envía al nodo 170 de transacción de unión y difunde un mensaje de objeto disponible a todos los demás nodos en el sistema. Por tanto, los átomos de catálogo maestro en cada nodo se actualizan y sincronizan.

Sin esperar la recepción del mensaje de objeto, el nodo de transacción de unión también puede comenzar el proceso de la etapa 185 para recuperar una copia del átomo de base de datos y el átomo gestor de transacción del nodo 172 seleccionado.

Cada uno de los demás nodos que reciben el mensaje de objeto disponible responde en la etapa 186 enviando un mensaje de acusado recibo de objeto al nodo seleccionado.

El motor de petición de base de datos en el nodo de unión inicia esta secuencia. Así, cuando se ha completado el

método de la figura 18, el nodo 170 de transacción de unión se conecta al sistema de bases de datos y, tal como se muestra en las figuras 4A y 4B, incluye una copia del átomo 70 de catálogo maestro, el átomo 71 gestor de transacción y el átomo 72 de base de datos. Después de eso, el nodo 170 de transacción de unión o bien creará o bien obtendrá copias de otros átomos según sea necesario.

Durante algunas operaciones, un nodo de transacción puede crear una nueva tabla, lo que requiere la creación de un nuevo átomo de tabla. La figura 20 da a conocer ese proceso 190 en el que el nodo A es un nodo solicitante, el átomo X es el átomo que solicita y gestionará un nuevo átomo Y. Para crear un nuevo átomo Y de tabla, el átomo X sería un átomo de esquema y el catálogo local Z sería El átomo de catálogo maestro. La etapa 191 representa funciones preparatorias para la petición de obtener una instancia del átomo Y. Cada vez que se cambia un átomo, se le asigna un número de cambio; el valor inicial normalmente es “0”.

En la etapa 192, el catálogo local Z crea esa instancia del átomo Y sin contenido y designa el nodo local como director para el átomo Y. Luego, un proceso 193 permite que el catálogo local Z asigne una ID de objeto al nuevo átomo Y. Los detalles de tal proceso se muestran y describen con respecto a la figura 21.

A continuación, el catálogo local Z establece un bloqueo exclusivo para permitir que se produzcan cambios sin ninguna influencia de la automatización externa. Mientras está impuesto el bloqueo, el catálogo local Z establece el estatus del nuevo átomo Y en un estatus “sucio” y en un estatus “no abierto”. El estatus “sucio” indica que el nuevo átomo Y no se ha reproducido en un archivo. El estatus “no abierto” indica que el nuevo átomo Y aún no está disponible para otros nodos. En la etapa 195, el catálogo local Z se actualiza y luego libera el bloqueo exclusivo. En la etapa 196, el catálogo Z difunde un mensaje de objeto disponible que identifica el átomo Y a todas las demás instancias del catálogo Z en los nodos de transacción y de archivado.

En la etapa 197, el átomo X, como el átomo de gestión para el nuevo átomo Y, rellena la instancia del átomo Y y establece el estatus para el átomo Y en “abierto”, lo que indica que el nuevo átomo puede reproducirse. Algún tiempo después de eso, un nodo de archivado guardará una copia del nuevo átomo en un almacenamiento persistente. Es decir, el nodo de archivado responderá a la recepción del mensaje de objeto disponible solicitando una copia del átomo Y para proporcionar de ese modo la redundancia. Cuando esto se complete, la recepción de un mensaje de acusado recibo de objeto desde el nodo de archivado hará que se cambie el estatus “sucio” y este cambio se reflejará entonces en todos los demás nodos que tienen una copia del átomo Y.

El proceso 193 de asignación en la figura 20 y procesos de asignación similares se usan en diversas etapas del funcionamiento de esta invención para asignar ID de objeto. La figura 21 representa este proceso de asignación con mayor detalle. Específicamente, cuando un átomo de catálogo desea asignar una ID de objeto, determina en la etapa 200 si tiene una ID local disponible. Si lo tiene, el control pasa a la etapa 201 que asigna la ID disponible en respuesta a la petición. Si no hay una ID local disponible, el control pasa a la etapa 202 para determinar si el átomo de catálogo en este nodo tiene un estado de “director”. Si es así, el director tiene la autoridad para identificar las ID disponibles directamente y controlar los pases a la etapa 201.

Tal como se recordará, cuando existe al menos otro nodo para el átomo en particular, cada átomo contiene una lista de nodos que incluyen copias del átomo que se solicita. Si esta fuera una primera petición para el átomo después de que ese átomo se hubiera creado según el proceso de la figura 20, el catálogo correspondiente en el primer nodo de la lista es el director. La etapa 203 representa el proceso de seleccionar un nodo e identificar al director. Para el proceso de selección, se prefiere establecer primero las comunicaciones con un nodo de transacción. La etapa 204 representa la transmisión de un mensaje de petición de ID al director.

Cuando el director recibe el mensaje de petición de ID en la etapa 205, obtiene un bloque de números de ID disponibles (etapa 206). El director identifica el nodo de transacción que realiza la petición como el sitio de este bloque de números de identificación disponibles (etapa 207). Luego, el director envía un mensaje de delegación de ID en la etapa 210 al nodo solicitante. Cuando el nodo solicitante recibe el mensaje de delegación de ID procedente del director en la etapa 211, almacena el bloque de números de ID en la etapa 212 y luego selecciona la primera ID disponible para la asignación en la etapa 201. Volviendo a hacer referencia a la figura 20, en esa situación, el proceso 193 se transfiere de la etapa 200 a la etapa 202 directamente a la etapa 201 porque el director se designa en la etapa 192 en la figura 20.

Siempre que el motor 41 de base de datos en las figuras 4A y 4B realizan una petición para un átomo, se procesa una respuesta 220 en la figura 22. Por ejemplo, supóngase que el motor 41 de petición de base de datos en un nodo 221 solicitante solicita una copia del átomo Y (por ejemplo, un átomo de tabla) de un nodo 222 seleccionado en una base de datos con otros nodos 223. Cuando el motor 41 de petición de base de datos realiza esa demanda, la etapa 224 determina si el átomo Y está presente en el nodo 221 solicitante. Si es así, la etapa 225 termina el proceso porque el átomo solicitado está presente. Si no es así, se transfiere el control a la etapa 226 mediante la que un catálogo local Z crea una instancia vacía del átomo Y y declara que el átomo Y no está “relleno”. Si el motor 41 de base de datos solicitara un átomo 74 de tabla, el átomo 70 de catálogo maestro realizaría esta etapa. La etapa 230 usa luego el proceso de selección descrito anteriormente para identificar un nodo 222 seleccionado. La preferencia es seleccionar cualquiera de los nodos de transacción antes de seleccionar el nodo de archivado de mayor

respuesta como el nodo 222 seleccionado.

En la etapa 231, el nodo 221 solicitante envía un mensaje de petición de objeto para el átomo Y al nodo 222 seleccionado. En respuesta, el nodo 222 seleccionado usa la etapa 232 para enviar un mensaje de objeto con el átomo Y solicitado en su forma serializada con los números de nodo y secuencia.

Al mismo tiempo, el nodo 222 seleccionado difunde un mensaje de objeto disponible a todos los demás nodos 223. También crea una lista de retransmisión para todos los demás nodos con una copia del átomo Y. En este punto del proceso, los demás nodos no se comunican directamente con el átomo Y en el nodo solicitante porque no saben que deben enviar mensajes de reproducción para ese átomo al nodo solicitante. Por tanto, cuando cualquiera de los demás nodos reproduce su átomo Y, el nodo 222 seleccionado retransmitirá el mensaje al nodo 221 solicitante.

Cuando el nodo 221 solicitante recibe el mensaje de objeto procedente del nodo 222 seleccionado, realiza un análisis de accesibilidad del mensaje en la etapa 233. Si el mensaje contiene un átomo actual, el átomo Y procesa el mensaje en la etapa 236 y envía un mensaje de acusado recibo de objeto al nodo seleccionado en la etapa 237.

Todos los demás nodos 223 usan la etapa 240 para responder a los mensajes de objeto disponible enviando un mensaje de acusado recibo de objeto al nodo 222 seleccionado. El nodo 222 seleccionado usa la etapa 241 para monitorizar los mensajes de acusado recibo de objeto. Específicamente, elimina cada uno de los otros nodos de su lista de retransmisión en respuesta a cada mensaje de acusado recibo de objeto y cesa las retransmisiones a ese nodo. Cuando todos los demás nodos 223 se han eliminado de la lista, el nodo seleccionado detiene toda transmisión y difunde un mensaje de objeto completo.

Es posible que otro nodo envíe un mensaje de reproducción que llegue al nodo 221 solicitante durante el tiempo entre las etapas 226 y 236 en la figura 22. Esto puede interrumpir el procesamiento de mensajes. Por consiguiente, cuando se recibe un mensaje de reproducción para el átomo Y en el ejemplo anterior, en el nodo solicitante, mientras el átomo Y no está relleno, se coloca en una lista de mensajes pendientes que forma parte de cada átomo, aunque no se muestra en ninguna de las figuras 5 a 13.

Tal como se indicó anteriormente, esta invención está adaptada particularmente a una base de datos que interacciona con técnicas de procesamiento de transacciones. Como tal, debe estar implicado un enfoque apropiado para los mensajes de “verificación”. La figura 23 representa un enfoque que garantiza la consistencia de los datos en tal entorno. Específicamente, la figura 23 representa cuatro nodos, a saber: el nodo 250 de transacción A, el nodo 251 de transacción B, el nodo 252 de transacción C y el nodo 253 de archivado. Se supone que el nodo 251 de transacción B envía un mensaje de verificación previa con una ID de transacción suministrada por el átomo gestor de transacción en ese nodo en la etapa 254. El mensaje se encamina al nodo de archivado en la etapa 255. Cuando se han completado todas las condiciones para verificar esa transacción, el nodo 255 de archivado emite un mensaje de transacción de verificación en la etapa 256 y, en la etapa 257, difunde ese mensaje de verificación. Cada uno de los nodos de transacción actualiza su número de transacción correspondiente en respuesta en la etapa 258.

Tal como se describió anteriormente con respecto al proceso para solicitar una copia de un átomo en la figura 21, el nodo solicitante realiza el análisis de accesibilidad sobre cualquier mensaje recibido en la etapa 233. Esta prueba garantiza que cualquier nodo de transacción siempre funciona con información válida e implica un análisis de números de ID de transacción y números de secuencia de verificación. La comprensión de este análisis y otras características de esta invención se verá facilitada por una mayor comprensión de las ID de transacción y los números de secuencia de verificación, mediante un análisis del orden relativo de las transacciones y mediante una comprensión del “sesgo atómico”.

Con respecto a las ID de transacción y los números de secuencia de verificación y, tal como se indicó anteriormente, cada identificador de transacción es único a través de todo el sistema de bases de datos de la figura 1. El propósito de una ID de transacción es ser una marca única, permanente y de todo el sistema que indica qué transacción creó una versión de registro específica. Cada ID de transacción se asigna mediante una copia local de un átomo 71 gestor de transacción y uno de tales átomos gestores de transacción será el director que asigna bloques de números de identificación a los átomos gestores de transacción en cada nodo de transacción en el sistema de bases de datos. Un átomo gestor de transacción en un nodo de transacción dado asigna un número no usado en un bloque asignado para cada transacción que inicia ese nodo. Como resultado, las ID de transacción de las transacciones más nuevas iniciadas en un nodo dado pueden ser mayores que las ID de transacción asignadas a transacciones más antiguas iniciadas en ese mismo nodo. Sin embargo, en una base de todo el sistema, las ID de transacción no implican nada sobre los tiempos de inicio relativos de las diferentes transacciones.

Por consiguiente, debe proporcionarse algún método para garantizar que una transacción pueda leer una versión de registro sólo si la transacción que creó la versión de registro se había verificado antes de que se iniciara la transacción de lectura. En esta invención, tal método para determinar qué versión de registro leer es el tiempo de verificación de transacción para la transacción que creó la versión de registro, no el tiempo de inicio. Una ID de transacción no contiene su tiempo de verificación, de modo que el átomo gestor de transacción en cada nodo asigna un número de secuencia de verificación a cada transacción basándose en la operación de verificación real. Cada

nodo de transacción genera sus números de secuencia de verificación en una secuencia creciente. Si una transacción de un nodo dado tiene un número de secuencia de verificación de 467, es seguro que se han verificado todas las transacciones de ese nodo con menores números de secuencia de verificación. Todos los nodos generan los mismos números de secuencia de verificación, de modo que interpretar los números requiere tanto el número como la identificación del nodo correspondiente.

Cuando se verifica una transacción, su átomo gestor de transacción envía un mensaje de verificación a todos los nodos que incluyen la ID de transacción y el número de secuencia de verificación correspondientes. Cuando un átomo se serializa a sí mismo, incluye el mayor número de secuencia de verificación que ha visto de cada nodo que contiene una copia de ese átomo. Por ejemplo, supóngase que los nodos A, B y C contienen copias de un átomo Z y que el nodo A genera un mensaje de objeto para el átomo Z. La forma serializada de ese mensaje incluirá el mayor número de secuencia de verificación que el nodo A ha visto de sí mismo y los mayores números de secuencia de verificación que ha visto de los nodos B y C. El mensaje serializado del nodo A describe el estado de las transacciones en todos los nodos que comparten una copia del átomo Y tal como lo ve el nodo A. Es posible que el nodo B o el nodo C hayan emitido en realidad mayores números de secuencia de verificación para las transacciones que el nodo A aún no ha recibido o procesado.

El gestor de transacción en cada nodo mantiene un objeto de transacción para cada nodo en el sistema de bases de datos. Cada objeto de transacción refleja el estado de las transacciones en todos los nodos y asigna a cada objeto dos números más que son locales y aumentan continuamente. Cuando el átomo gestor de transacción inicia una transacción o recibe un mensaje de transición y transacción que indica que otro nodo ha iniciado una transacción, el átomo gestor de transacción crea un objeto de transacción local con la nueva ID de transacción y le asigna un número de inicio. Cuando se verifica una transacción local o cuando el átomo gestor de transacción recibe un mensaje de transición y transacción que indica que se ha verificado una transacción en otro nodo, se asigna un número de fin de transacción al objeto de transacción junto con el número de secuencia de verificación. Los números de inicio y fin proceden de la misma secuencia. Como ejemplo, si la transacción 123 tiene un número de inicio que es mayor que el número de fin de transacción 453, la transacción 123 puede leer las versiones de registro creadas por la transacción 453 desde cualquier nodo que ejecutó la transacción 453.

Los números de inicio y fin de transacción son locales; es decir, reflejan el estado de las transacciones tal como se ven en el nodo local. Los diferentes nodos asignan diferentes valores y ven un ordenamiento diferente de las transacciones en otros nodos. Tal como resultará evidente, pueden existir retardos en la recepción y el procesamiento de los mensajes de estado de transacción. Cada transacción se ejecuta sólo en su nodo de transacción local. Este sistema de bases de datos impide que un nodo de transacción “vea” una transacción como verificada antes de que la transacción se verifique en su nodo de transacción local. Por ejemplo, si el nodo A inicia la transacción 123 después de recibir y procesar el mensaje de verificación procedente del nodo de transacción B para la transacción 453, la transacción 123 puede leer los cambios de la transacción 453, aunque un nodo de transacción C que recibió la transacción verifique e inicie mensajes en otro orden y considere que las dos transacciones son contemporáneas.

Como ninguna de las transacciones se ejecuta en el nodo de transacción C, la información no es relevante para el nodo de transacción C. Sin embargo, la diferencia entre la visión del sistema de bases de datos por cada nodo de transacción individual puede provocar problemas en determinadas circunstancias.

Con respecto al “sesgo atómico”, según un aspecto de esta invención, cada nodo debe procesar mensajes de otro nodo en el orden en que el otro nodo envió los mensajes. Sin embargo, todos los nodos funcionan de forma independiente. Por tanto, en cualquier tiempo dado, algunos nodos habrán recibido y procesado mensajes que otros nodos aún no han procesado. Algunas copias de átomos incluirán cambios que otras copias no. Esas diferencias no importan para cualquier nodo que tenga una copia porque cada copia del átomo está en un estado consistente para ese nodo. Tal como se describió anteriormente, siempre que un nodo de transacción cambia un átomo, el nodo reproduce el átomo modificado en todos los demás nodos que contienen una copia de ese átomo. Si los nodos de transacción B y C contienen, cada uno, una copia de ese átomo y el nodo de transacción B cambia ese átomo, el nodo de transacción B envía mensajes de reproducción al nodo de transacción C. Mientras el nodo de transacción B está procesando una transacción, envía un mensaje de estado de transacción al nodo de transacción C y reproduce cualquier átomo que cambie el nodo de transacción B. Cuando el nodo de transacción B verifica la transacción mediante el proceso que se muestra en la figura 23, el nodo de transacción C debería haber recibido el mensaje de estado de transacción que indica que la transacción se ha verificado. El nodo de transacción C procesará todos los mensajes relacionados con la transacción antes de que reciba el mensaje de verificación debido al hecho de que todos los mensajes se procesarán en el orden en que fueron enviados. Las copias de átomos en diferentes nodos de transacción pueden diferir, pero todos los cambios realizados por una transacción en cualquier nodo de transacción se realizarán en todos los demás nodos antes de que el otro nodo vea la transacción como verificada.

Al ver las transacciones desde el nivel del sistema, los diferentes nodos de transacción no actúan en sincronismo. Los tiempos para que un único mensaje sea procesado por diferentes nodos pueden variar debido a las diferencias en la eficiencia de la trayectoria de comunicaciones y la cantidad de mensajes que es necesario que procese cada nodo de transacción. Deben tenerse en cuenta las diferencias en los tiempos de procesamiento en diferentes nodos.

Considérense tres nodos de transacción: nodo A, nodo B y nodo C. Supóngase que el nodo B ejecuta una transacción 768 que cambia el átomo X y que el nodo C tiene una copia del átomo X. Supóngase que el átomo X en el nodo B envía mensajes de cambio al nodo C y que hay un lapso de tiempo antes de que el nodo C procese esos cambios. Supóngase también que el nodo B ha asignado un número de secuencia de verificación de 47 a la transacción 768. El átomo gestor de transacción en el nodo B envía un mensaje de estado de transacción a todos los nodos en el sistema de bases de datos. También supóngase que el nodo A solicita una copia del átomo X del nodo C después de recibir el mensaje de verificación procedente del nodo B, pero antes de que el nodo C complete el procesamiento de la transacción. El nodo A recibe y procesa el mensaje de estado de transacción procedente del nodo B. Desde la perspectiva del nodo A, se verifica la transacción 768 y el mayor número de secuencia de verificación para el nodo B es 47. Sin embargo, con esta temporización, el átomo devuelto desde el nodo C no refleja los cambios realizados por la transacción en el nodo B.

Según esta invención, cuando el átomo X en el nodo C se serializa a sí mismo para su transmisión al nodo A, incluye el mayor número de secuencia de verificación de cada nodo que tiene una copia del átomo X. En este caso, supóngase que el mensaje incluiría un número de secuencia de verificación 97 para el nodo C y 46 para el nodo B.

Haciendo referencia a la figura 22, el nodo A como nodo solicitante crea una instancia vacía del átomo X antes de pedirle al nodo C su contenido. El análisis de accesibilidad en la figura 22, obtiene el mayor número de secuencia de verificación actual para cada nodo tal como se ve en el nodo A. También expande el mensaje serializado con el átomo X para obtener los mayores números de secuencia de verificación actuales para cada nodo tal como se ve en el nodo B. En esta comparación, el análisis de accesibilidad determinará que el número de secuencia de verificación serializado en el mensaje es 46, mientras que su mayor número de secuencia de verificación local para B es 47. Por tanto, el nodo A no puede usar la versión serializada del átomo X en su estado actual y debe esperar a que el átomo X en nodo C le envíe un mensaje de objeto completo. Si el número de secuencia de verificación es igual o mayor, el nodo A puede continuar procesando el átomo.

Tal como se indicó anteriormente, cuando el nodo C envió la copia serializada del átomo X al nodo A, envió un mensaje de objeto disponible a otras copias del átomo X, en este caso, a la copia en el nodo B. El átomo X en el nodo B envía un mensaje de acusado recibo de objeto al átomo X en el nodo C y añade la copia en el nodo A a su lista de copias de sí mismo. Después de eso, el átomo X en el nodo B enviará mensajes de cambio a sus copias en ambos nodos A y C. Mientras tanto, el átomo X en el nodo C procesa los mensajes procedentes del nodo B, retransmitiéndolos al nodo A. El átomo X en el nodo A procesa los mensajes, volviéndose cada vez más actualizado. Cuando el átomo X en el nodo C procesa el mensaje de acusado recibo de objeto del átomo X en el nodo B y todas las demás copias del átomo X en otros nodos, envía un mensaje de objeto completo al átomo X en el nodo A.

Según esta invención, cada nodo de transacción debe funcionar sin tener que transferir ningún átomo al disco u otro almacenamiento persistente. Esto requiere la eliminación de átomos inactivos de cada nodo de transacción de manera rápida. Para lograr este objetivo, una clase de elemento de ayuda en cada nodo proporciona una función de recopilación de elementos inválidos llamada periódicamente por un gestor de ciclos para iniciar un nuevo ciclo de recopilación de elementos inválidos independientemente de otras operaciones. Más específicamente, el gestor de ciclos realiza un ciclo de vencimiento mediante el que cualquier átomo al que se hace referencia en un ciclo anterior se mueve al frente en una lista de uso menos reciente (LRU, *least recently used*) en el átomo de catálogo maestro para el nodo. El gestor de ciclos obtiene un bloqueo exclusivo para garantizar que no haya otros subprocesos activos con respecto a un ciclo anterior.

Luego, el gestor de ciclos inicia el subproceso de recopilación de elementos inválidos de registro para un átomo de tabla que está marcado para la recopilación de elementos inválidos de registro. Si la cantidad actual de memoria en el nodo supera una cantidad especificada definida por un objeto de configuración identificado por el puntero 70U en la figura 5, el proceso de subprocesos de recopilación de elementos inválidos pasa por la LRU de catálogo maestro desde los átomos de uso menos reciente hasta los de uso más reciente. Clasifica un átomo como candidato para un candidato de recopilación de elementos inválidos si ese átomo: (1) no está activo, (2) no se ha hecho referencia en el ciclo de vencimiento actual, (3) no es un átomo de “objeto incompleto”, (4) se ha cambiado localmente y aún no se ha escrito en un nodo de archivado y (5) es un catálogo que no tiene átomos residentes en la memoria. Un átomo en un nodo de archivado adicionalmente no debe haberse serializado en el disco.

Si el proceso de recopilación de elementos inválidos determina que se han cumplido todas las condiciones anteriores, hay dos opciones. Si el nodo es un nodo de archivado y hay otras instancias del átomo en otros nodos, el contenido del átomo se “purga”. De lo contrario, se solicita que el átomo se suprima a sí mismo, lo que hace comprobando primero con su átomo de catálogo de creación para una prueba final. Si se pasa esa prueba, el átomo de catálogo borra su puntero para el átomo candidato que difunde un mensaje de objeto no disponible a otros nodos que contienen instancias del átomo candidato y luego se suprime a sí mismo. Este ciclo continúa átomo a átomo hasta que la memoria de trabajo esté dentro de los límites aceptables.

Hay dos mecanismos implicados en este proceso. Cuando se le pide a un átomo de catálogo que encuentre un átomo para la clasificación, el átomo de catálogo establece su entrada de referencia de ciclo en el número de ciclo actual. Además, los átomos gestores de transacción, de base de datos, de esquema, de secuencia y de tabla

obtienen un bloqueo compartido en un ciclo actual para que los métodos en esos átomos puedan contener punteros de átomo sin tener que incrementar el recuento de uso de un átomo para impedir que un objeto o un átomo sea recopilado como elementos inválidos.

Las figuras 19 y 20 comentan la “dirección”. Tal como se indica en la figura 20 cuando se crea un átomo, el nodo de creación se designa como director de átomo y establece una lista ordenada de nodos para el átomo. Cuando un nodo crea un átomo, es la única entrada en la lista ordenada. El primer nodo en la lista es el director.

Cuando otro nodo solicita más tarde una copia de ese átomo, el nodo seleccionado coloca la identificación del nodo solicitante en la lista ordenada justo después de sí mismo, ya sea el director o no. Como resultado, si el nodo designado como director de un átomo se vuelve inactivo por cualquier motivo, todas las demás copias de ese átomo tienen la lista ordenada de nodos. Al revisar la lista ordenada, el próximo director es el primer nodo de transacción restante en la lista. Si no hay nodos de transacción en la lista, se designa el primer nodo de archivado que no es de sincronización.

Ahora puede ser útil como ayuda para comprender la interacción entre diferentes nodos y átomos dentro de un nodo para comentar una consulta simple de base de datos. Supóngase que el motor de petición de base de datos en un nodo de transacción recibe una consulta de base de datos para seleccionar todos los registros en una tabla “miembro” con un código postal de “01944”. Se supone que el átomo gestor de transacción y el átomo de tabla están ubicados en el nodo local, que se ha completado todo el procesamiento de consulta y que existe un átomo de índice para un campo “postal” en la tabla “miembro”.

Inicialmente, el motor 41 de petición de base de datos en la figura 2 emite una orden al átomo gestor de transacción para comenzar una nueva transacción, después de lo cual el átomo gestor de transacción asigna una ID de transacción. A continuación, el motor de petición de base de datos usa el átomo de tabla para la tabla “miembro” para determinar si existe un átomo de índice para el campo “postal”. Si estos dos átomos están en el nodo local, se procesan. Si no lo están, se obtienen copias de estos átomos de otros nodos antes de que continúe el procesamiento.

El átomo de tabla para la tabla “miembro” utiliza información del átomo de catálogo de tablas correspondiente para explorar el índice. Esto produce un mapa de bits que identifica cada “registro” en la tabla “miembro” con el código postal especificado. Tal como resultará evidente, este mapa de bits puede estar limitado a un campo particular o tal vez el resultado de la combinación lógica de índices de múltiples campos.

A continuación, se establece un bucle basado en el mapa de bits resultante. Para cada iteración del bucle, el motor de petición de base de datos emite una llamada al átomo de tabla para procesar un método de registro de captura especificado con la ID de la transacción en el registro. El átomo de tabla selecciona un átomo de estados de registro apropiado que está relacionado con un átomo de registro dividiendo el número de registro entre un número fijo que corresponde al número máximo de ID de registro que gestiona un átomo de estados de registro. A continuación, el átomo de tabla usa un método de captura en el átomo de estados de registro seleccionado usando la ID de la transacción y el número de registro identificado. Para registros multiversión, el átomo de estados de registro pasa por cualquier versión de registro para encontrar la versión correcta y el puntero apropiado para el átomo de datos correspondiente. El átomo de estados de registro llama a ese átomo de datos con el número de átomo de datos que apunta al registro y permite su recuperación. Cuando se completa este proceso, el motor de petición de base de datos proporciona al usuario un conjunto de registros que enumera todos los registros con el código postal especificado.

En resumen, resultará evidente para los expertos en la técnica que un sistema de gestión de bases de datos construido según esta invención proporciona un sistema de procesamiento de datos elástico, variable a escala, bajo demanda, distribuido. El sistema es tolerante a fallos y tiene un alto grado de disponibilidad. Es independiente de la plataforma y funciona para proporcionar una base de datos atómica, consistente, aislada y duradera. Además, puede funcionar a través de Internet sin la necesidad de trayectorias de comunicaciones de alta velocidad y está adaptado para el procesamiento de transacciones que puede implementarse en un área geográfica amplia.

Esta invención logra todos estos objetivos implementando una o más de las siguientes características. Esta invención fragmenta la base de datos en objetos distribuidos que se reproducen con una base de igual a igual. Cada nodo de transacción y de archivado determina qué átomos deben residir en la memoria de manera local. Los átomos de catálogo realizan un seguimiento de ubicaciones para copias locales y remotas de diversos átomos e identifican los catálogos de los que son miembros. Además, cada nodo puede determinar el mejor de múltiples nodos para solicitar una copia del átomo que permite sistemas dispersos geográficamente.

REIVINDICACIONES

1. Sistema de gestión de base de datos que permite que los usuarios interaccionen con una base de datos que se compone de datos y metadatos, comprendiendo dicho sistema:

una pluralidad de nodos de transacción que proporcionan a los usuarios acceso a la base de datos, y al menos un nodo de archivado que mantiene un archivo de toda la base de datos, incluyendo cada nodo de transacción:

i) un motor de petición de base de datos que proporciona una interfaz entre órdenes de consulta de entrada y salida de alto nivel a nivel de usuario y órdenes de entrada y salida a nivel del sistema que controlan una secuencia de operaciones para interaccionar con la base de datos, en el que, en respuesta a determinadas órdenes a nivel del sistema, objetos atómicos generan átomos, y

ii) una memoria no persistente que almacena al menos una copia de un átomo en un conjunto de átomos, conteniendo cada átomo un fragmento especificado de datos o metadatos, definiendo colectivamente el conjunto de todos los átomos todos los metadatos y datos en la base de datos,

una red de sistema de bases de datos que interconecta todos los nodos, y

un control de comunicaciones en cada uno de los nodos para establecer una trayectoria de comunicaciones con cada uno de los demás nodos en el sistema, en el que:

cada nodo de transacción está configurado, respondiendo a una orden de sistema procedente del motor de petición de base de datos, para solicitar una copia de un átomo que es relevante para la orden de consulta pero que no está presente en ese nodo de transacción de un nodo de transacción que tiene una copia del átomo solicitado,

cada nodo de transacción está configurado, respondiendo a una petición de una copia de un átomo de otro nodo de transacción, para reproducir el átomo solicitado para la transferencia al nodo de transacción solicitante mediante lo cual sólo es necesario que los átomos que se requiere que completen una orden de consulta estén ubicados en algún nodo de transacción en cualquier tiempo dado, y

cada nodo de transacción está configurado, respondiendo a un cambio en un átomo en ese nodo de transacción, para reproducir ese cambio en cualquier otro nodo en el sistema que contenga una copia de ese átomo.

2. Sistema de gestión de base de datos según la reivindicación 1, en el que los átomos de catálogo en un nodo de transacción realizan un seguimiento de:

i) todos los átomos que son residentes en ese nodo de transacción, y

ii) la ubicación de cada nodo que tiene una copia de cada uno de los átomos residentes.

3. Sistema de gestión de base de datos según la reivindicación 2, en el que uno de los átomos de catálogo es un átomo de catálogo maestro que se reproduce en cada nodo conectado al sistema, en el que el conjunto de átomos comprende un átomo gestor de transacción para la base de datos para realizar un seguimiento del progreso de transacciones iniciadas por la interfaz, y en el que el catálogo maestro identifica la ubicación del átomo gestor de transacción.

4. Sistema de gestión de base de datos según la reivindicación 3, en el que la base de datos a nivel de usuario comprende tablas designadas con registros y campos y funciona según al menos un esquema designado y en el que un conjunto de átomos que son residentes en un nodo de transacción comprende:

i) un átomo de tabla para cada tabla que contiene una lista de campos,

ii) un átomo de esquema que establece una correspondencia entre un esquema y las tablas designadas relevantes para ese esquema, y

iii) un átomo de base de datos que establece una correspondencia entre la base de datos y el esquema designado.

5. Sistema de gestión de base de datos según la reivindicación 4, en el que las tablas de base de datos a nivel de usuario comprende registros identificados que contienen campos y datos, en el que el conjunto de todos los átomos comprende átomos de datos que contienen un número predeterminado de registros de datos y un átomo de estados de registro que gestiona versiones de registro y en el que dicho átomo de estados de

registro incluye una matriz que identifica los átomos de datos gestionados asignados a ese átomo de estados de registro, una identificación de cada versión de registro y, para cada versión de registro, información sobre esa versión y la ubicación del átomo de datos que contiene esa versión.

6. Sistema de gestión de base de datos según la reivindicación 1, en el que cada nodo de transacción está configurado además para:

i) generar y recibir mensajes predefinidos en respuesta a órdenes a nivel del sistema, y

ii) seleccionar una de las trayectorias de comunicaciones para transmitir el mensaje.

7. Sistema de gestión de base de datos según la reivindicación 6, en el que cada nodo de transacción está configurado para para someter a prueba de manera asíncrona la eficiencia de cada trayectoria de comunicaciones.

8. Sistema de gestión de base de datos según la reivindicación 7, en el que cada nodo de transacción está configurado para someter a prueba la eficiencia generando un mensaje de *ping* y recibiendo un mensaje de acuse de recibo de *ping* para determinar el tiempo de *ping* para cada trayectoria de comunicaciones.

9. Sistema de gestión de base de datos según la reivindicación 6, en el que cada nodo de transacción está configurado para serializar un átomo en un mensaje y deserializar el mensaje en un átomo y en el que cada mensaje se envía de manera asíncrona por la trayectoria de comunicaciones seleccionada con un protocolo con controles que mantienen secuencias de mensajes.

10. Sistema de gestión de base de datos según la reivindicación 1, en el que nodos individuales pueden estar activos y conectados al sistema y pueden desconectarse del sistema, uno de los átomos es un átomo de catálogo maestro que identifica cada nodo activo en el sistema y que se reproduce en cada nodo conectado al sistema, y el sistema comprende:

i) medios para establecer una conexión entre un nodo de transacción de unión y un nodo seleccionado de los nodos activos,

ii) medios en el nodo de transacción de unión para pedir al nodo seleccionado la conexión al sistema,

iii) medios en el nodo seleccionado para actualizar su átomo de catálogo maestro para indicar el nodo de transacción de unión como nodo activo,

iv) medios en el nodo seleccionado para transferir un mensaje al nodo de transacción de unión que incluye la identificación del nodo seleccionado y para difundir un mensaje que indica la disponibilidad del nodo de transacción de unión a todos los demás nodos en el sistema,

v) medios en dicho nodo de transacción de unión para responder al mensaje solicitando una copia del átomo de catálogo maestro del nodo seleccionado, y

vi) medios en el nodo seleccionado para transferir su átomo de catálogo maestro actualizado al nodo de transacción de unión.

11. Sistema de gestión de base de datos según la reivindicación 10, en el que dicho nodo de transacción de unión incluye medios que responden a la recepción del átomo de catálogo maestro actualizado para solicitar copias de átomos adicionales en respuesta a órdenes de sistema procedentes de su motor de petición de base de datos.

12. Sistema de gestión de base de datos según la reivindicación 10, en el que el nodo seleccionado difunde la disponibilidad de catálogo maestro actualizado a todos los demás nodos y cada uno de los demás nodos incluye medios para dar acuse de recibo de su recepción del catálogo maestro actualizado al nodo seleccionado.

13. Sistema de gestión de base de datos según la reivindicación 1, en el que el motor de petición de base de datos en uno de los nodos de transacción realiza una petición que requiere la producción de un nuevo átomo, comprendiendo adicionalmente dicho nodo de transacción un átomo de catálogo para:

i) realizar un seguimiento del nuevo átomo que responde a la petición de motor de petición de base de datos,

ii) asignar una identificación de objeto al nuevo átomo,

iii) establecer un estatus para el nuevo átomo que indica que no hay una copia redundante del nuevo átomo y que el nuevo átomo no está disponible para otros nodos, y

iv) difundir a todos los demás átomos de catálogo correspondientes en el sistema la existencia del nuevo átomo.

14. Sistema de gestión de base de datos según la reivindicación 13, en el que el al menos un nodo de archivado está configurado, respondiendo a la recepción del nuevo átomo, para enviar un mensaje al nuevo átomo que actualiza su estatus para indicar la existencia de una copia redundante.

15. Sistema de gestión de base de datos según la reivindicación 1, en el que un motor de petición de base de datos en un nodo de transacción solicita un átomo que sólo existe en otros nodos y en el que:

i) un átomo de catálogo solicitante está configurado para realizar un seguimiento del átomo solicitado,

ii) el átomo de catálogo solicitante está configurado para seleccionar el nodo con mayor respuesta y solicitar una copia del átomo del nodo con mayor respuesta,

iii) el nodo seleccionado está configurado para enviar una copia del átomo solicitado en un mensaje de objeto al nodo solicitante y para difundir la disponibilidad del átomo a otros nodos en los que reside el átomo,

iv) el nodo solicitante está configurado para evaluar la validez del mensaje de objeto recibido, rellenar el átomo vacío, y reenviar un mensaje de acuse de recibo de objeto al nodo seleccionado.

16. Sistema de gestión de base de datos según la reivindicación 15, en el que dicho nodo seleccionado genera una lista de retransmisión de todos los nodos que contienen el átomo solicitado a los que el nodo seleccionado reenviará mensajes relacionados con ese átomo al nodo solicitante.

17. Sistema de gestión de base de datos según la reivindicación 16, en el que cada uno de los demás nodos que reciben el mensaje de disponibilidad de objeto difundido está configurado para enviar un mensaje de acuse de recibo de objeto al nodo seleccionado, actualizando dicho nodo seleccionado la lista de retransmisión en respuesta a la recepción de cada mensaje de acuse de recibo de objeto desde otro nodo para terminar la retransmisión de cualquier mensaje posterior al nodo solicitante.

18. Sistema de gestión de base de datos según la reivindicación 1, adaptado para garantizar la consistencia de la base de datos durante el procesamiento de transacciones en el que el sistema incluye un primer nodo de transacción que implica que se verifique una transacción y al menos uno de los demás nodos de transacción que incluye una copia de ese átomo, incluyendo dicho sistema:

i) el primer nodo de transacción está configurado para generar un mensaje de verificación previa con una identificación de transacción para la transferencia a un nodo de archivado,

ii) respondiendo al mensaje de verificación previa, el nodo de archivado está configurado para verificar la transacción y después de eso, difundir un mensaje de verificación con la identificación de transacción a todos los nodos, y

iii) el primer nodo de transacción y el al menos uno de los demás nodos están configurados, respondiendo a la recepción del mensaje de verificación, para actualizar la transacción identificada.

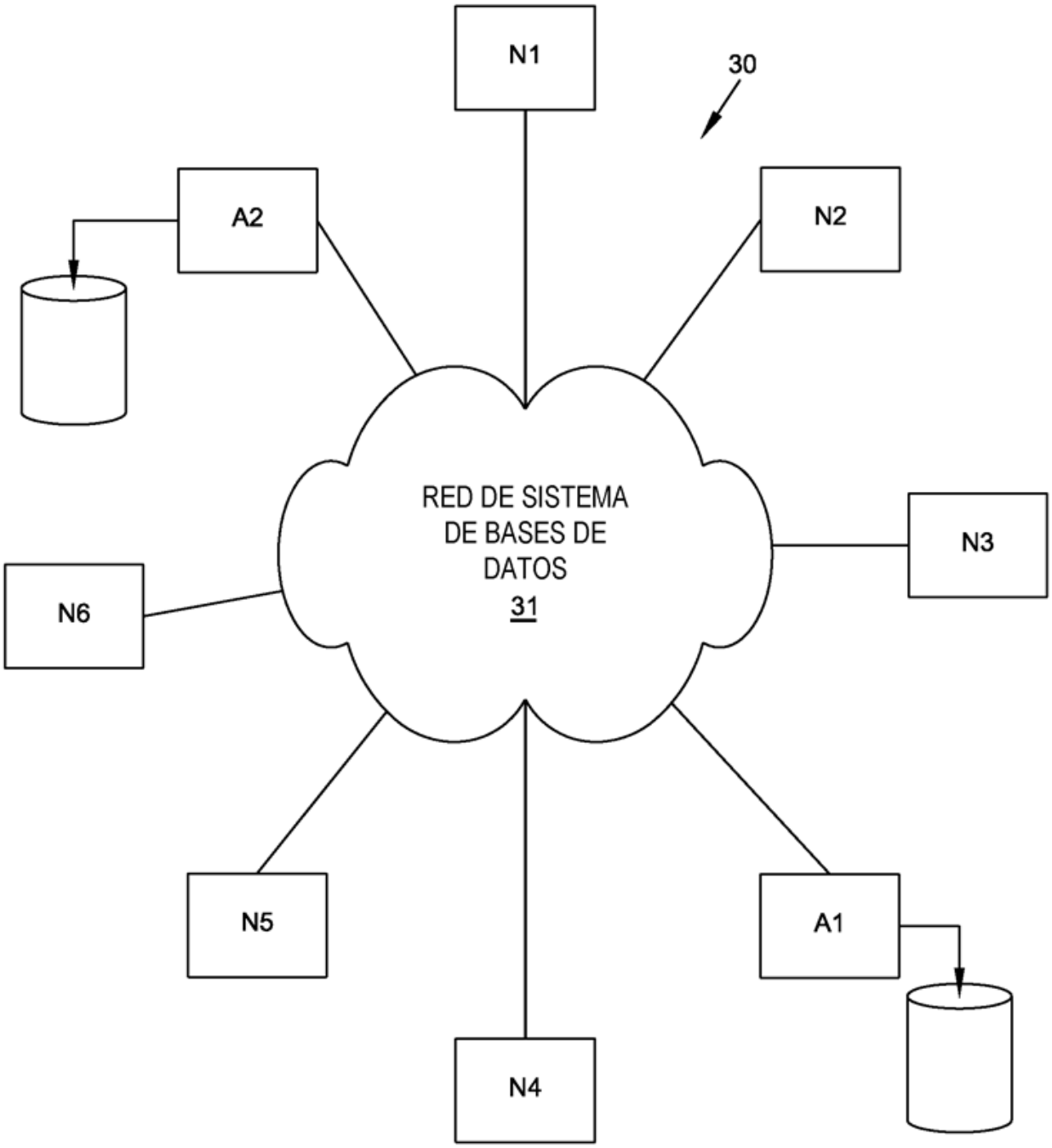

FIG. 1

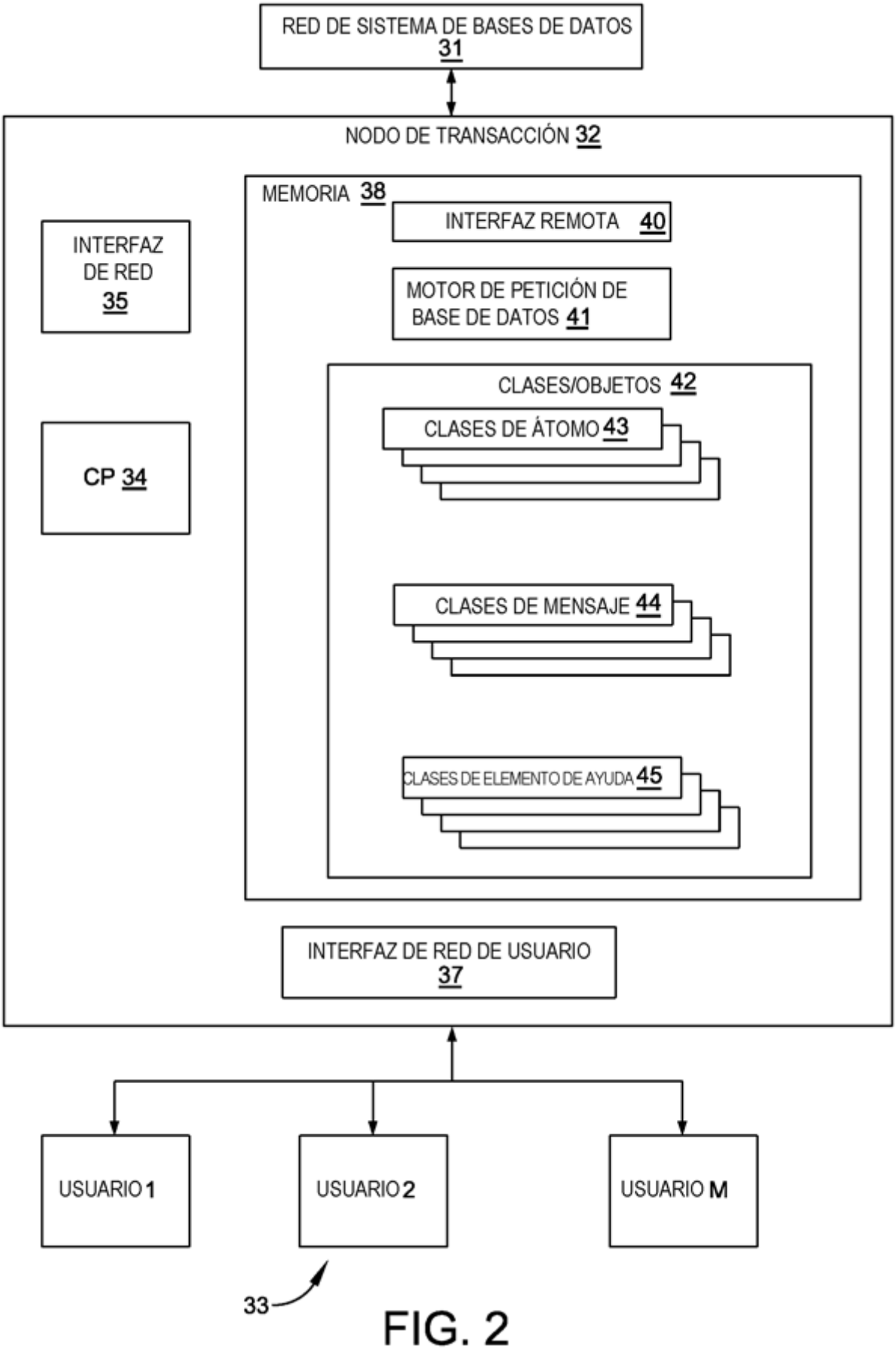
RED DE SISTEMA DE BASES DE DATOS 31
NODO DE TRANSACCIÓN 32
MEMORIA 38
INTERFAZ REMOTA 40
MOTOR DE PETICIÓN DE BASE DE DATOS 41
CLASES/OBJETOS 42
CLASES DE ÁTOMO 43
CLASES DE MENSAJE 44
CLASES DE ELEMENTO DE AYUDA 45
INTERFAZ DE RED 35
CP 34
INTERFAZ DE RED DE USUARIO 37
USUARIO 1
USUARIO 2
USUARIO M
33

FIG. 2

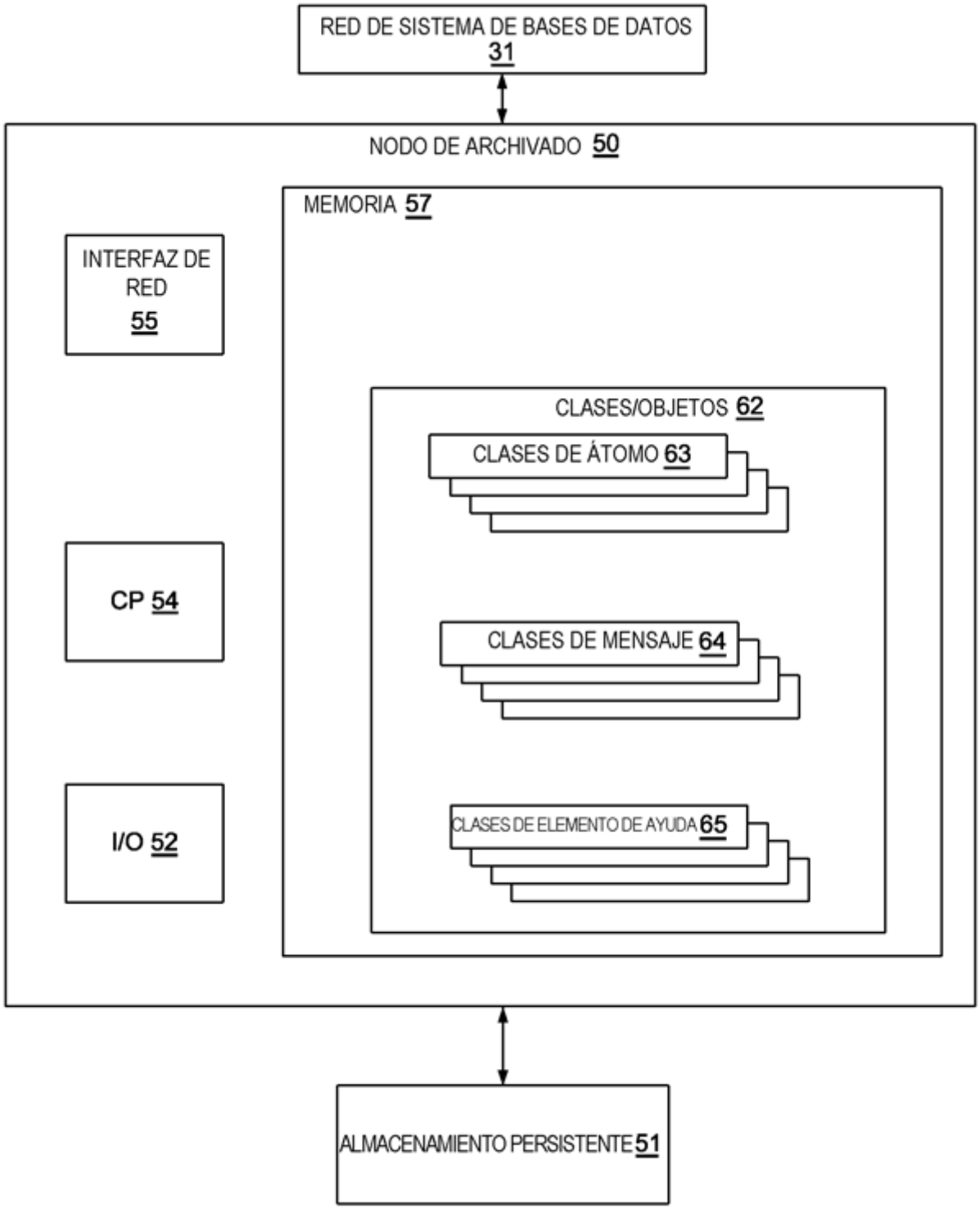
RED DE SISTEMA DE BASES DE DATOS
31
NODO DE ARCHIVADO 50
MEMORIA 57
INTERFAZ DE RED
55
CLASES/OBJETOS 62
CLASES DE ÁTOMO 63
CP 54
CLASES DE MENSAJE 64
I/O 52
CLASES DE ELEMENTO DE AYUDA 65
ALMACENAMIENTO PERSISTENTE 51

FIG. 3

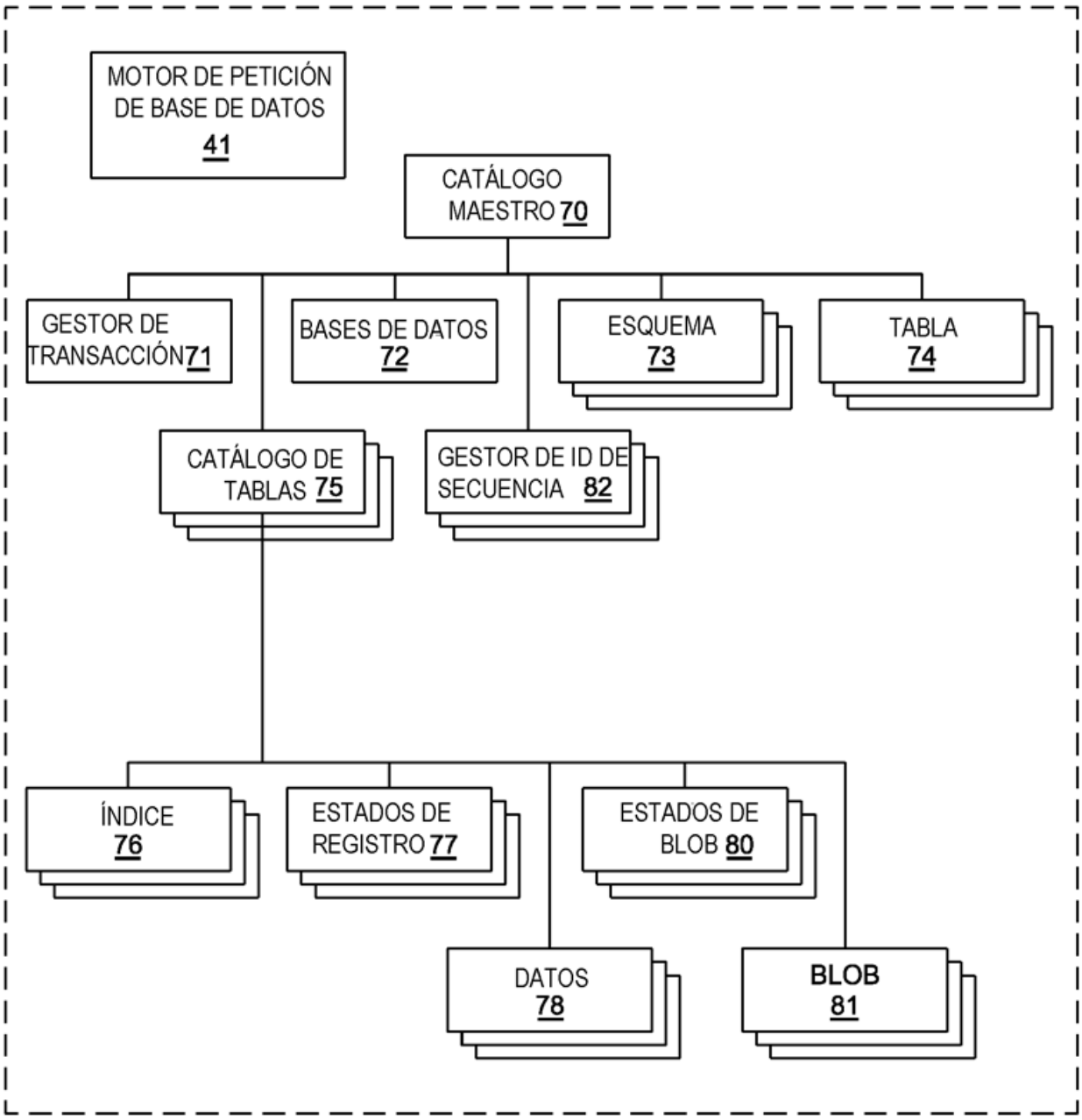
MOTOR DE PETICIÓN DE BASE DE DATOS 41
CATÁLOGO MAESTRO 70
GESTOR DE TRANSACCIÓN 71
BASES DE DATOS 72
ESQUEMA 73
TABLA 74
CATÁLOGO DE TABLAS 75
GESTOR DE ID DE SECUENCIA 82
ÍNDICE 76
ESTADOS DE REGISTRO 77
ESTADOS DE BLOB 80
DATOS 78
BLOB 81

FIG. 4A

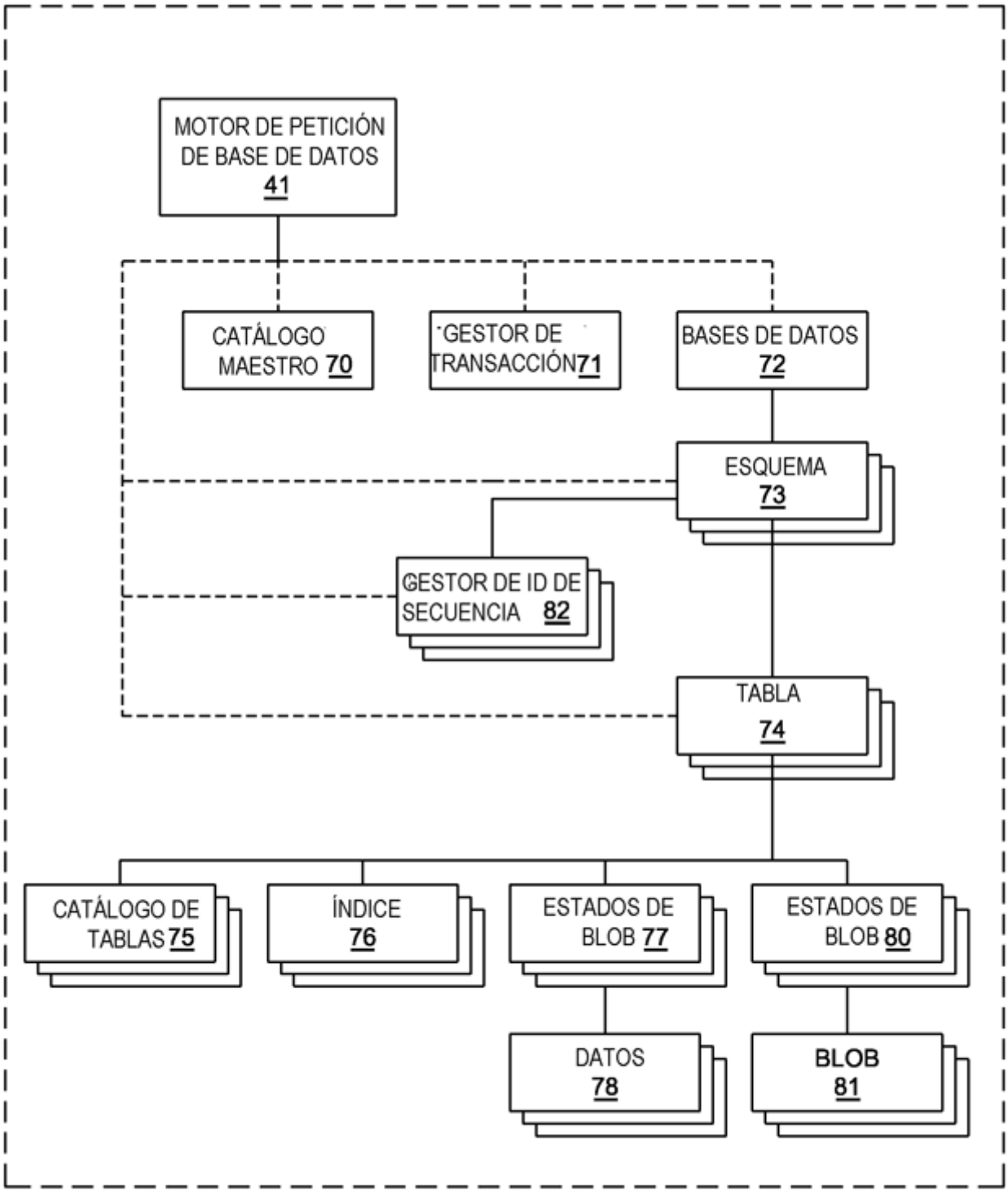
MOTOR DE PETICIÓN DE BASE DE DATOS 41
CATÁLOGO MAESTRO 70
GESTOR DE TRANSACCIÓN 71
BASES DE DATOS 72
ESQUEMA 73
GESTOR DE ID DE SECUENCIA 82
TABLA 74
CATÁLOGO DE TABLAS 75
ÍNDICE 76
ESTADOS DE BLOB 77
ESTADOS DE BLOB 80
DATOS 78
BLOB 81

FIG. 4B


ÁTOMO DE CATÁLOGO MAESTRO 70
70A
ID DE CATÁLOGO MAESTRO=1
70B
PUNTERO PARA CATÁLOGO MAESTRO
70C
PUNTERO PARA CATÁLOGO DE CREACIÓN
70D
PUNTERO PARA DIRECTOR
70E
NÚMERO DE CAMBIO
70F
LISTA DE RETRANSMISIONES
70G
REFERENCIA DE CICLO
70H
LISTA DE NODOS ACTIVOS
70I
ESTADOS DE ESTATUS
70J
ID DE NODO GLOBAL
70K
ID DE NODO LOCAL
70L
GESTOR DE CONEXIÓN
70M
NODO PENDIENTE
70N
UUI DE BASE DE DATOS
70P
CATÁLOGO DE BASES DE DATOS, ESQUEMAS, TABLAS Y ÁTOMOS DE CATÁLOGO DE TABLAS
70Q
CONTRASEÑA
70R
VERSIÓN REAL
70S
VERSIÓN DE SOFTWARE
70T
PUNTERO PARA ÁTOMO GESTOR DE TRANSACCIÓN
70U
PUNTERO PARA OBJETO DE CONFIGURACIÓN
70V
PUNTERO PARA SUBPROCESO DE RECOPILACIÓN DE ELEMENTOS INVÁLIDOS
70W
PUNTERO PARA SUBPROCESO DE PING

FIG. 5

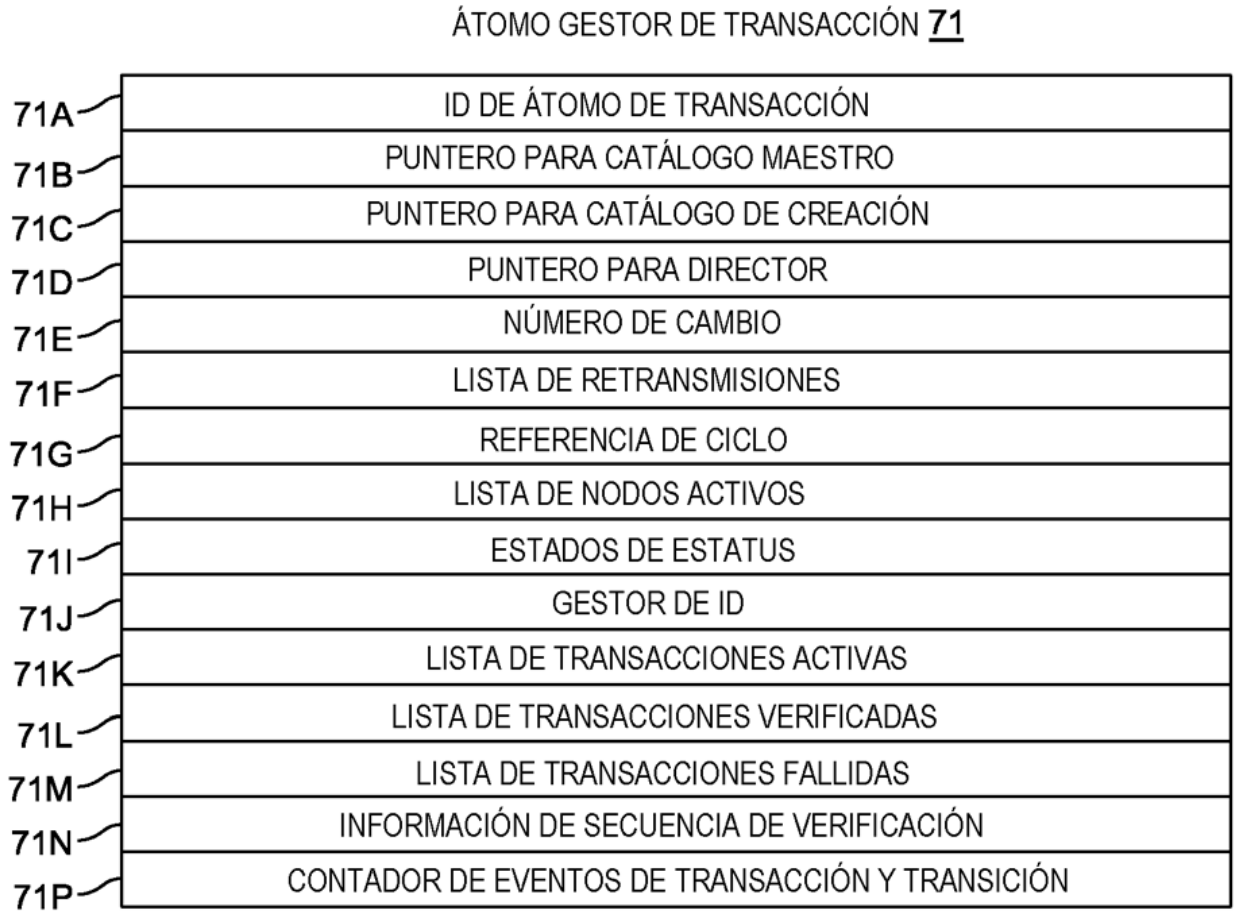
ÁTOMO GESTOR DE TRANSACCIÓN 71
71A
ID DE ÁTOMO DE TRANSACCIÓN
71B
PUNTERO PARA CATÁLOGO MAESTRO
71C
PUNTERO PARA CATÁLOGO DE CREACIÓN
71D
PUNTERO PARA DIRECTOR
71E
NÚMERO DE CAMBIO
71F
LISTA DE RETRANSMISIONES
71G
REFERENCIA DE CICLO
71H
LISTA DE NODOS ACTIVOS
71I
ESTADOS DE ESTATUS
71J
GESTOR DE ID
71K
LISTA DE TRANSACCIONES ACTIVAS
71L
LISTA DE TRANSACCIONES VERIFICADAS
71M
LISTA DE TRANSACCIONES FALLIDAS
71N
INFORMACIÓN DE SECUENCIA DE VERIFICACIÓN
71P
CONTADOR DE EVENTOS DE TRANSACCIÓN Y TRANSICIÓN

FIG. 6

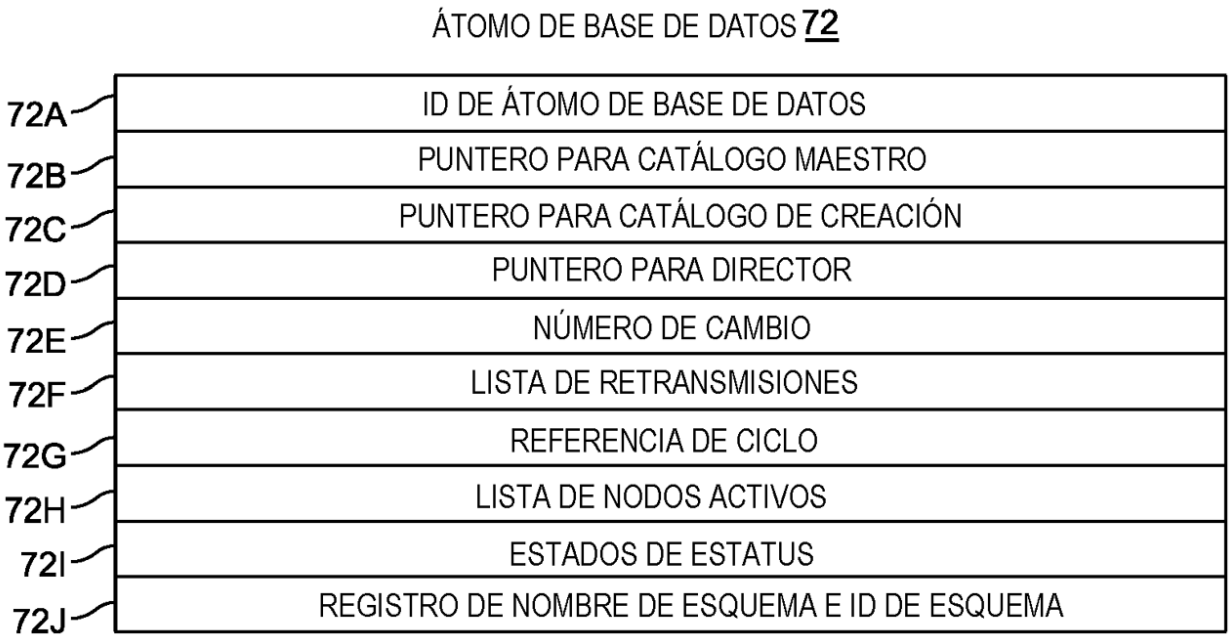
ÁTOMO DE BASE DE DATOS 72
72A
ID DE ÁTOMO DE BASE DE DATOS
72B
PUNTERO PARA CATÁLOGO MAESTRO
72C
PUNTERO PARA CATÁLOGO DE CREACIÓN
72D
PUNTERO PARA DIRECTOR
72E
NÚMERO DE CAMBIO
72F
LISTA DE RETRANSMISIONES
72G
REFERENCIA DE CICLO
72H
LISTA DE NODOS ACTIVOS
72I
ESTADOS DE ESTATUS
72J
REGISTRO DE NOMBRE DE ESQUEMA E ID DE ESQUEMA

FIG. 7

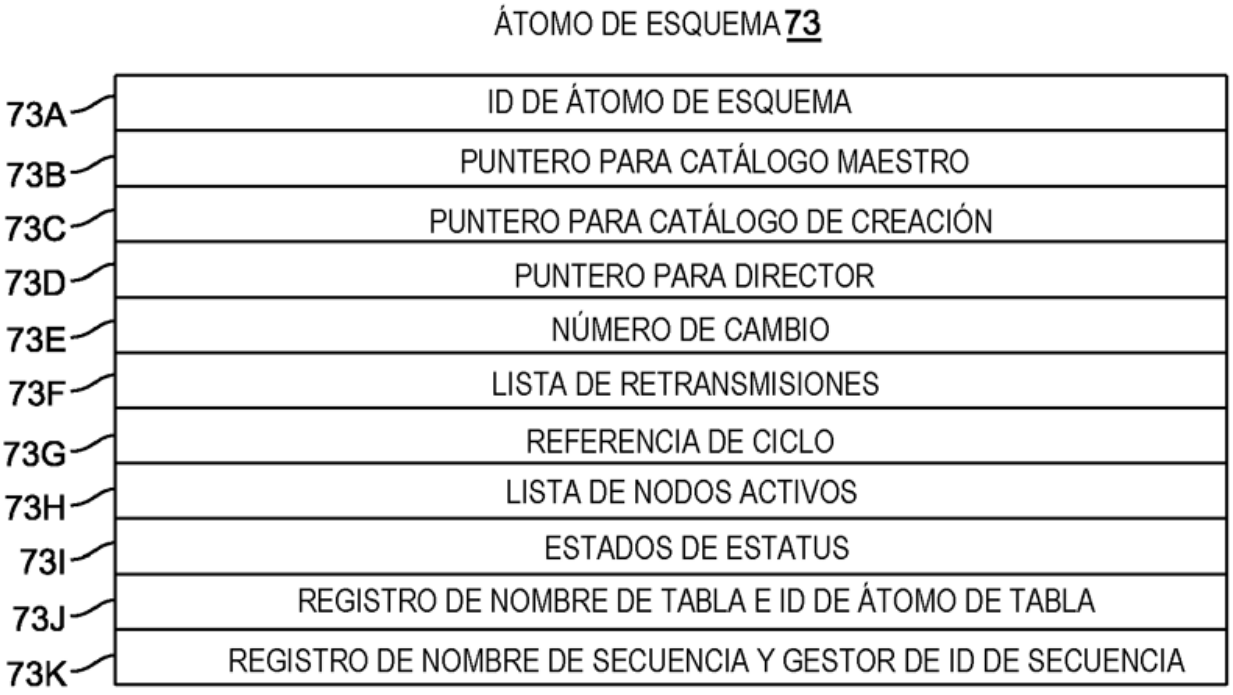
ÁTOMO DE ESQUEMA 73
73A
ID DE ÁTOMO DE ESQUEMA
73B
PUNTERO PARA CATÁLOGO MAESTRO
73C
PUNTERO PARA CATÁLOGO DE CREACIÓN
73D
PUNTERO PARA DIRECTOR
73E
NÚMERO DE CAMBIO
73F
LISTA DE RETRANSMISIONES
73G
REFERENCIA DE CICLO
73H
LISTA DE NODOS ACTIVOS
73I
ESTADOS DE ESTATUS
73J
REGISTRO DE NOMBRE DE TABLA E ID DE ÁTOMO DE TABLA
73K
REGISTRO DE NOMBRE DE SECUENCIA Y GESTOR DE ID DE SECUENCIA

FIG. 8

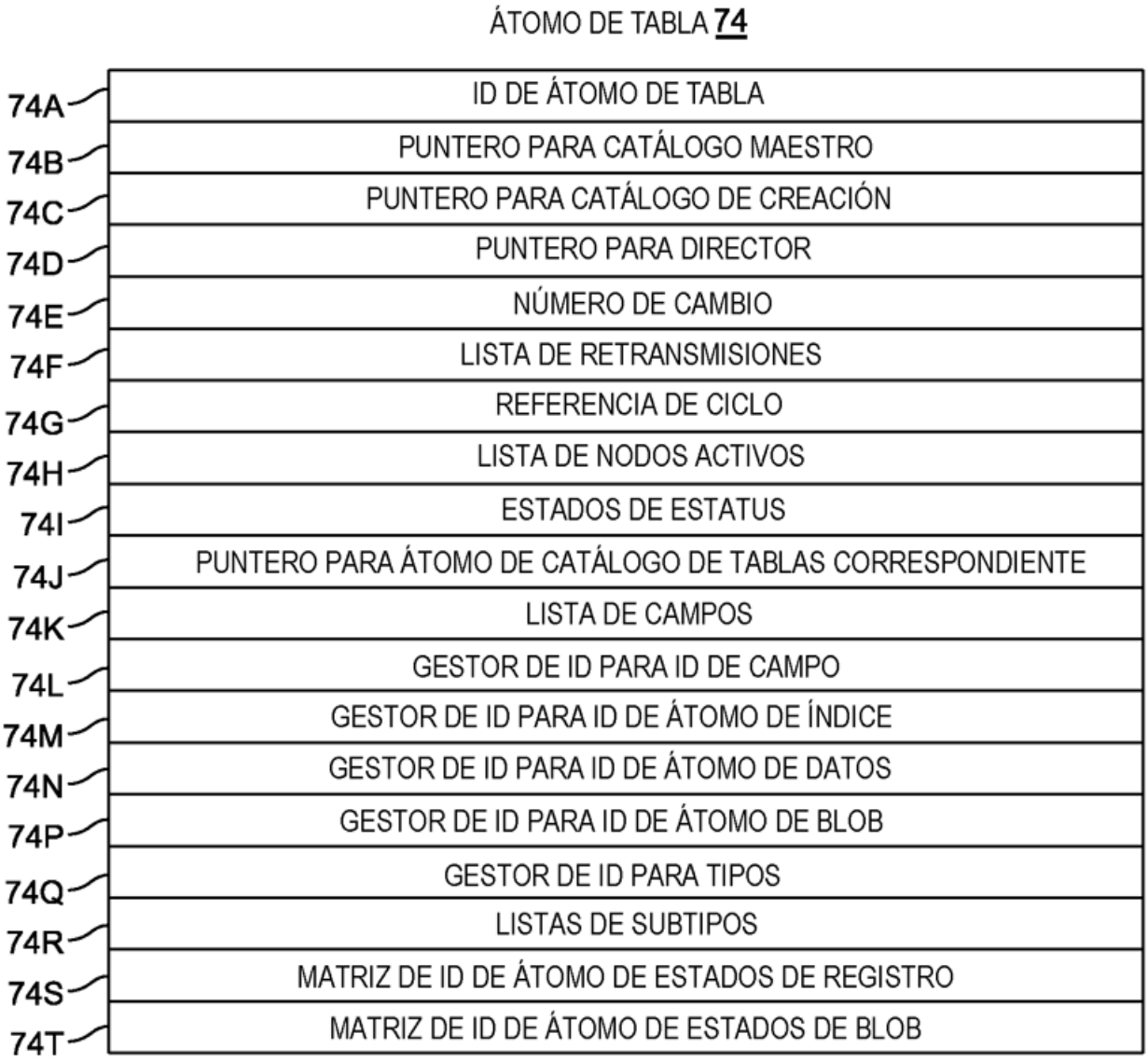
ÁTOMO DE TABLA 74
74A
ID DE ÁTOMO DE TABLA
74B
PUNTERO PARA CATÁLOGO MAESTRO
74C
PUNTERO PARA CATÁLOGO DE CREACIÓN
74D
PUNTERO PARA DIRECTOR
74E
NÚMERO DE CAMBIO
74F
LISTA DE RETRANSMISIONES
74G
REFERENCIA DE CICLO
74H
LISTA DE NODOS ACTIVOS
74I
ESTADOS DE ESTATUS
74J
PUNTERO PARA ÁTOMO DE CATÁLOGO DE TABLAS CORRESPONDIENTE
74K
LISTA DE CAMPOS
74L
GESTOR DE ID PARA ID DE CAMPO
74M
GESTOR DE ID PARA ID DE ÁTOMO DE ÍNDICE
74N
GESTOR DE ID PARA ID DE ÁTOMO DE DATOS
74P
GESTOR DE ID PARA ID DE ÁTOMO DE BLOB
74Q
GESTOR DE ID PARA TIPOS
74R
LISTAS DE SUBTIPOS
74S
MATRIZ DE ID DE ÁTOMO DE ESTADOS DE REGISTRO
74T
MATRIZ DE ID DE ÁTOMO DE ESTADOS DE BLOB

FIG. 9

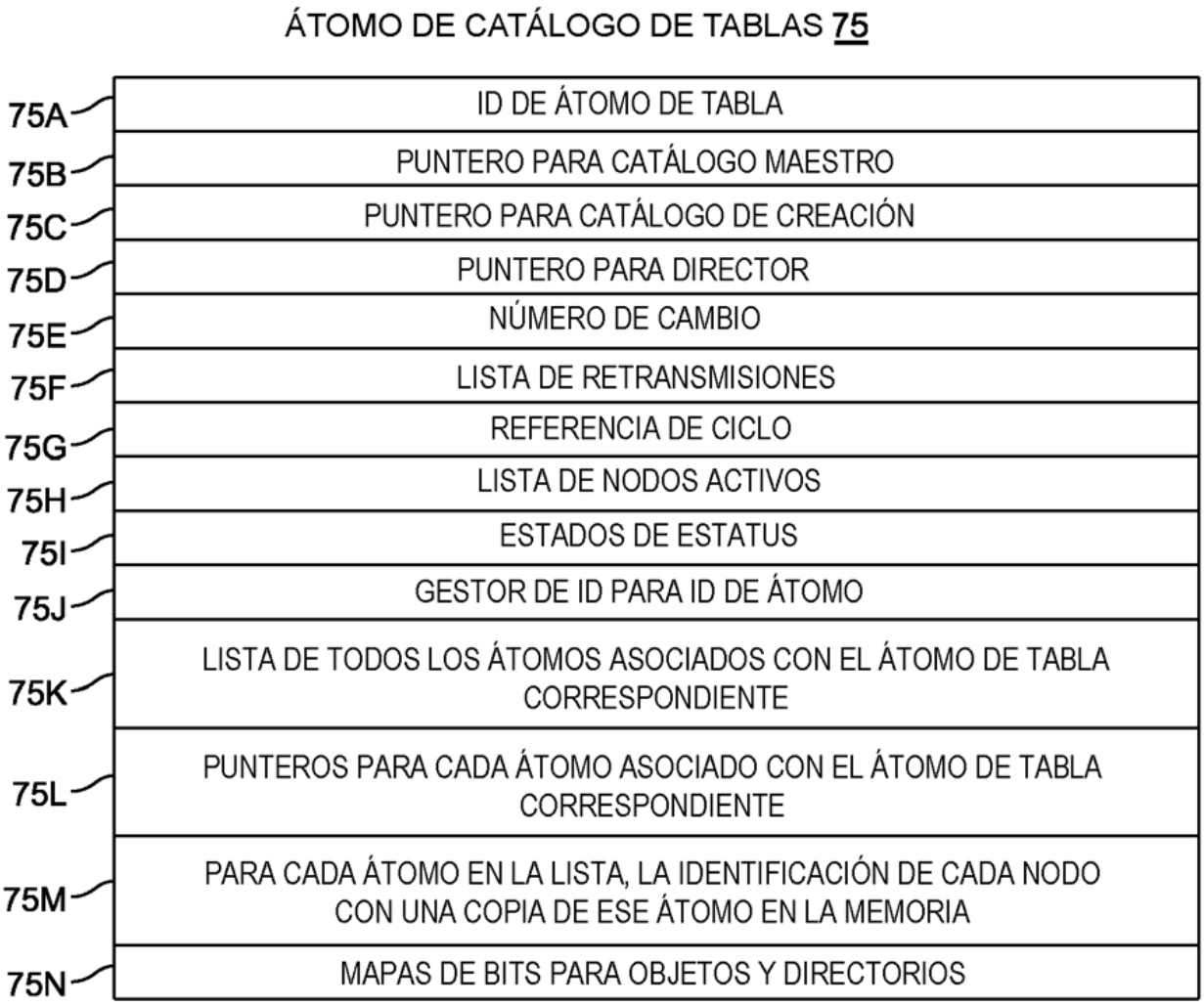
ÁTOMO DE CATÁLOGO DE TABLAS 75
75A
ID DE ÁTOMO DE TABLA
75B
PUNTERO PARA CATÁLOGO MAESTRO
75C
PUNTERO PARA CATÁLOGO DE CREACIÓN
75D
PUNTERO PARA DIRECTOR
75E
NÚMERO DE CAMBIO
75F
LISTA DE RETRANSMISIONES
75G
REFERENCIA DE CICLO
75H
LISTA DE NODOS ACTIVOS
75I
ESTADOS DE ESTATUS
75J
GESTOR DE ID PARA ID DE ÁTOMO
75K
LISTA DE TODOS LOS ÁTOMOS ASOCIADOS CON EL ÁTOMO DE TABLA CORRESPONDIENTE
75L
PUNTEROS PARA CADA ÁTOMO ASOCIADO CON EL ÁTOMO DE TABLA CORRESPONDIENTE
75M
PARA CADA ÁTOMO EN LA LISTA, LA IDENTIFICACIÓN DE CADA NODO CON UNA COPIA DE ESE ÁTOMO EN LA MEMORIA
75N
MAPAS DE BITS PARA OBJETOS Y DIRECTORIOS

FIG. 10

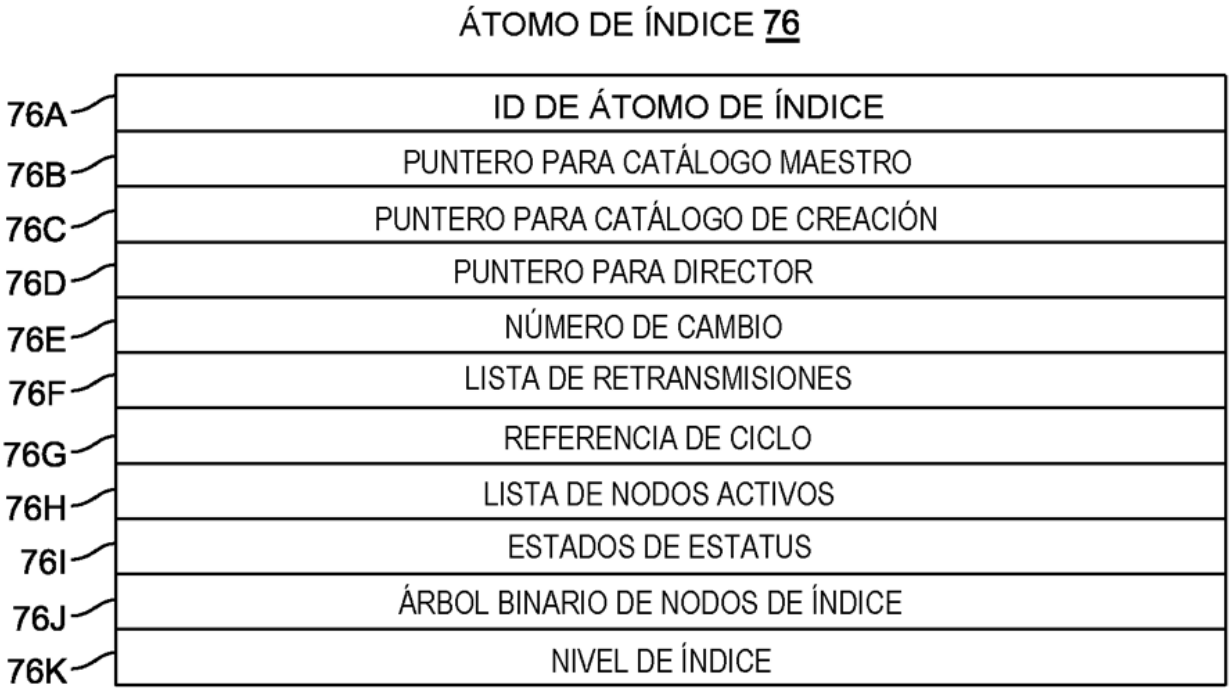
ÁTOMO DE ÍNDICE 76
76A
ID DE ÁTOMO DE ÍNDICE
76B
PUNTERO PARA CATÁLOGO MAESTRO
76C
PUNTERO PARA CATÁLOGO DE CREACIÓN
76D
PUNTERO PARA DIRECTOR
76E
NÚMERO DE CAMBIO
76F
LISTA DE RETRANSMISIONES
76G
REFERENCIA DE CICLO
76H
LISTA DE NODOS ACTIVOS
76I
ESTADOS DE ESTATUS
76J
ÁRBOL BINARIO DE NODOS DE ÍNDICE
76K
NIVEL DE ÍNDICE

FIG. 11


ÁTOMO DE ESTADOS DE REGISTRO 77
77A
ID DE ÁTOMO DE ESTADOS DE REGISTRO
77B
PUNTERO PARA CATÁLOGO MAESTRO
77C
PUNTERO PARA CATÁLOGO DE CREACIÓN
77D
PUNTERO PARA DIRECTOR
77E
NÚMERO DE CAMBIO
77F
LISTA DE RETRANSMISIONES
77G
REFERENCIA DE CICLO
77H
LISTA DE NODOS ACTIVOS
77I
ESTADOS DE ESTATUS
77J
MATRIZ DE ÁTOMOS DE DATOS GESTIONADOS
77K
ID DE REGISTRO PARA REGISTRO BASE
77L
PUNTERO PARA ÁTOMO DE TABLA
77M
MAPA DE BITS DE REGISTROS CON VERSIÓN
77N
PARA CADA VERSIÓN DE REGISTRO
77P
ID DE TRANSACCIÓN
77Q
VERSIÓN DE FORMATO
77R
NÚMERO DE SECUENCIA DE VERSIÓN DE REGISTRO
77S
UBICACIÓN DE SIGUIENTE VERSIÓN MÁS ANTIGUA
77T
ÍNDICE PARA MATRIZ DE ÁTOMOS DE DATOS
77U
RANURA EN ÁTOMO DE DATOS PARA DATOS REALES

FIG. 12


ÁTOMO DE DATOS 78
78A
ID DE ÁTOMO DE DATOS
78B
PUNTERO PARA CATÁLOGO MAESTRO
78C
PUNTERO PARA CATÁLOGO DE CREACIÓN
78D
PUNTERO PARA DIRECTOR
78E
NÚMERO DE CAMBIO
78F
LISTA DE RETRANSMISIONES
78G
REFERENCIA DE CICLO
78H
LISTA DE NODOS ACTIVOS
78I
ESTADOS DE ESTATUS
78J
GESTOR DE ID PARA RANURAS DE REGISTRO
78K
DIRECCIÓN Y LONGITUD DE CADA REGISTRO DENTRO DEL ÁTOMO DE DATOS
78L
REGISTROS DE DATOS

FIG. 13

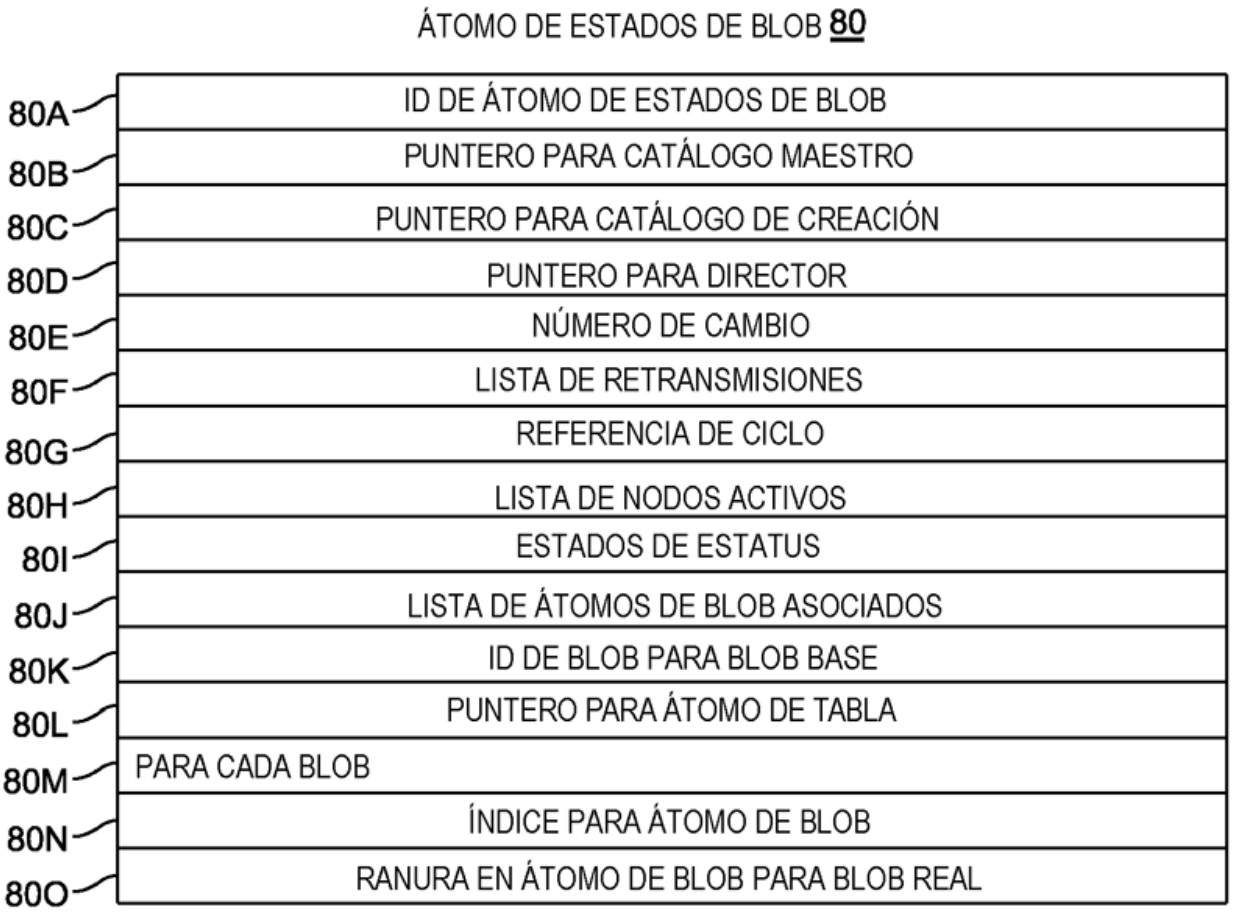
ÁTOMO DE ESTADOS DE BLOB 80
80A
ID DE ÁTOMO DE ESTADOS DE BLOB
80B
PUNTERO PARA CATÁLOGO MAESTRO
80C
PUNTERO PARA CATÁLOGO DE CREACIÓN
80D
PUNTERO PARA DIRECTOR
80E
NÚMERO DE CAMBIO
80F
LISTA DE RETRANSMISIONES
80G
REFERENCIA DE CICLO
80H
LISTA DE NODOS ACTIVOS
80I
ESTADOS DE ESTATUS
80J
LISTA DE ÁTOMOS DE BLOB ASOCIADOS
80K
ID DE BLOB PARA BLOB BASE
80L
PUNTERO PARA ÁTOMO DE TABLA
80M
PARA CADA BLOB
80N
ÍNDICE PARA ÁTOMO DE BLOB
80O
RANURA EN ÁTOMO DE BLOB PARA BLOB REAL

FIG. 14

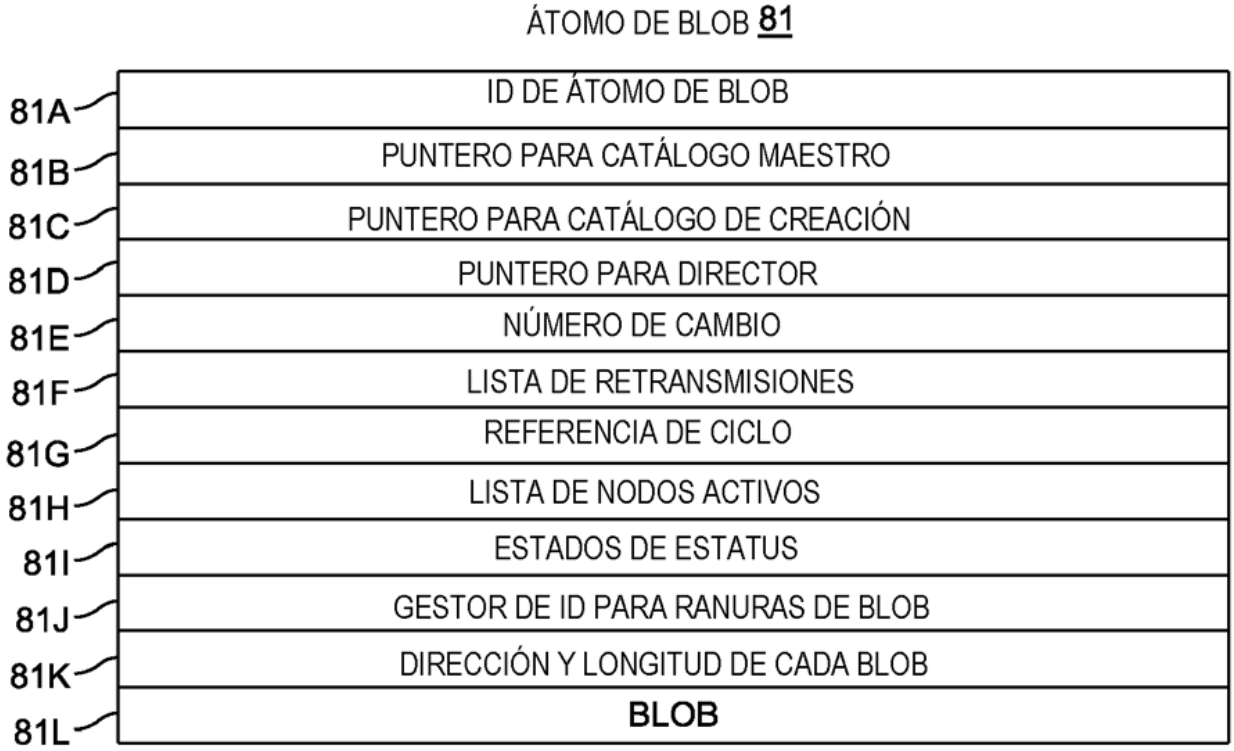
ÁTOMO DE BLOB 81
81A
ID DE ÁTOMO DE BLOB
81B
PUNTERO PARA CATÁLOGO MAESTRO
81C
PUNTERO PARA CATÁLOGO DE CREACIÓN
81D
PUNTERO PARA DIRECTOR
81E
NÚMERO DE CAMBIO
81F
LISTA DE RETRANSMISIONES
81G
REFERENCIA DE CICLO
81H
LISTA DE NODOS ACTIVOS
81I
ESTADOS DE ESTATUS
81J
GESTOR DE ID PARA RANURAS DE BLOB
81K
DIRECCIÓN Y LONGITUD DE CADA BLOB
81L
BLOB

FIG. 15

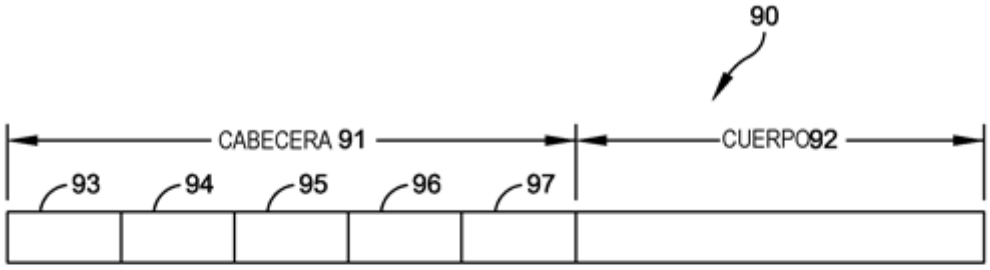

FIG. 16

MENSAJES

110	PING	135	OBJETO DE BLOB DE TABLA
111	ACUSE DE RECIBO DE PING	136	BLOB DE TABLA
112	CONECTAR	137	TIPO DE TABLA
113	BIENVENIDA	138	BORRAR REGISTRO DE TABLA
114	NUEVO NODO	139	RECOPILAR ELEMENTOS INVÁLIDOS DE TABLA
115	ESTADO DE NODO	140	PETICIÓN DE ACTUALIZACIÓN DE REGISTROS
116	PETICIÓN DE OBJETO	141	RESPUESTA DE ACTUALIZACIÓN DE REGISTROS
117	OBJETO	142	ACTUALIZACIÓN DE REGISTROS
118	OBJETO DISPONIBLE	143	OBJETO DE DATOS DE REGISTROS
119	OBJETO CON ACUSE DE RECIBO	144	REGISTRO DE REGISTROS
120	OBJETO COMPLETO	145	ELIMINACIÓN DE REGISTROS
121	OBJETO NO DISPONIBLE	146	REGISTRO DE ANULACIÓN
122	OBJETO DE DEVOLUCIÓN	147	SOLICITUD DE DIVISIÓN DE ÍNDICE
123	REGISTRAR OBJETO	148	DIVISIÓN DE ÍNDICE
124	CANCELAR REGISTRO DE OBJETO	149	ELIMINACIÓN DE ÍNDICE
125	SUPRIMIR OBJETO	150	NODO DE ÍNDICE AÑADIDO
126	OBJETO ESCRITO	151	ÍNDICE DE TABLA PREPARADO
127	PETICIÓN DE ID	152	BLOB DE BLOB
128	DELEGACIÓN DE ID	153	REGISTRO DE DATOS
129	CAMPO DE TABLA AÑADIDO	154	ESTADO DE TRANSACCIÓN
130	FORMATO DE TABLA	155	BLOQUEO DE TRANSACCIÓN
131	REGISTROS DE PETICIÓN DE TABLA	156	INTERBLOQUEO DE TRANSACCIÓN
132	OBJETO DE REGISTROS DE TABLA	157	TIEMPOS DE ESCRITURA DE PETICIÓN
133	REGISTRO DE TABLA	158	TIEMPOS DE ESCRITURA
134	ÍNDICE DE TABLA AÑADIDO		

FIG. 17

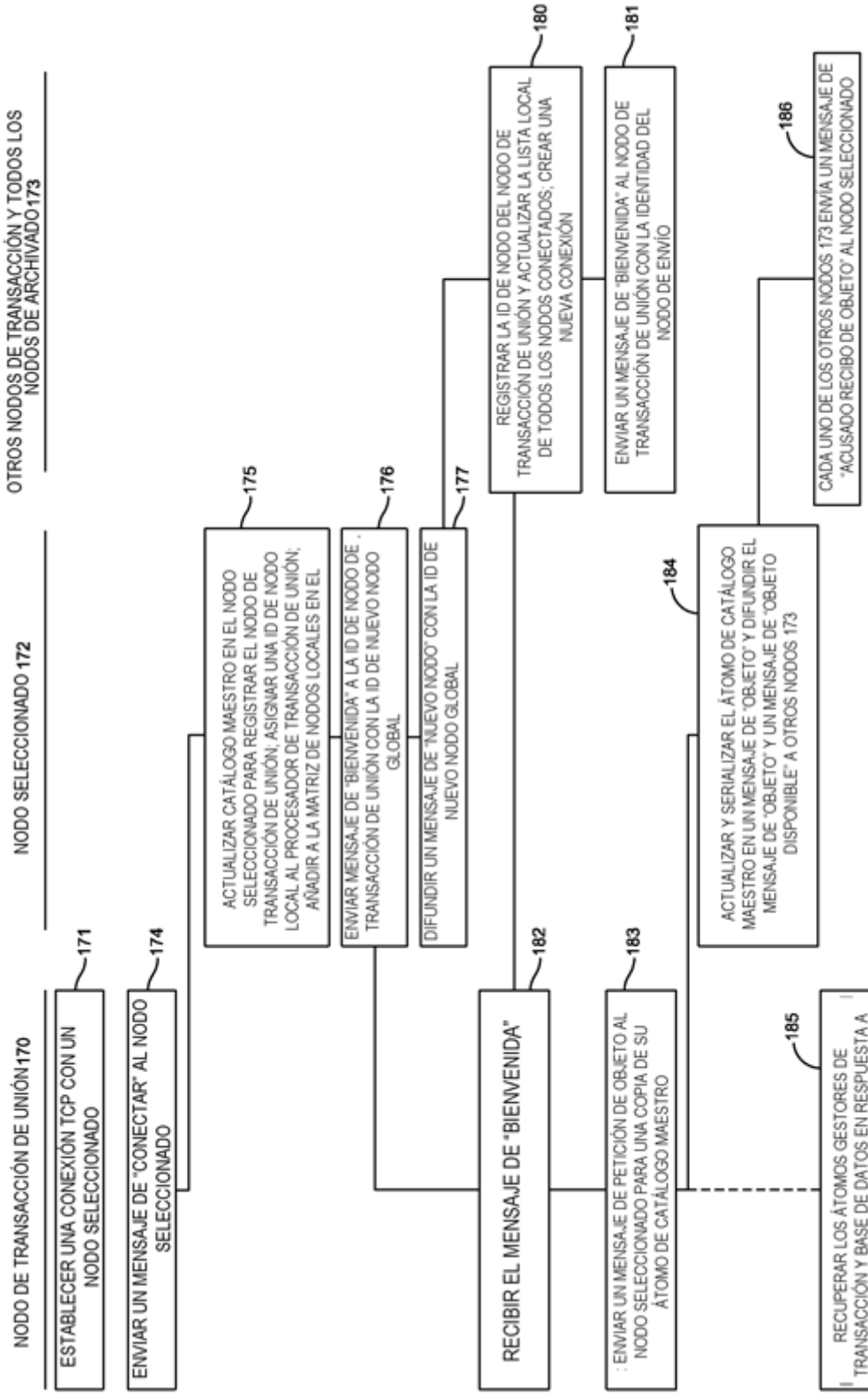
NODO DE TRANSACCIÓN DE UNIÓN 170
NODO SELECCIONADO 172
OTROS NODOS DE TRANSACCIÓN Y TODOS LOS NODOS DE ARCHIVADO 173
171
ESTABLECER UNA CONEXIÓN TCP CON UN NODO SELECCIONADO
174
ENVIAR UN MENSAJE DE "CONECTAR" AL NODO SELECCIONADO
175
ACTUALIZAR CATÁLOGO MAESTRO EN EL NODO SELECCIONADO PARA REGISTRAR EL NODO DE TRANSACCIÓN DE UNIÓN; ASIGNAR UNA ID DE NODO LOCAL AL PROCESADOR DE TRANSACCIÓN DE UNIÓN; AÑADIR A LA MATRIZ DE NODOS LOCALES EN EL
176
ENVIAR MENSAJE DE "BIENVENIDA" A LA ID DE NODO DE TRANSACCIÓN DE UNIÓN CON LA ID DE NUEVO NODO GLOBAL
177
DIFUNDIR UN MENSAJE DE "NUEVO NODO" CON LA ID DE NUEVO NODO GLOBAL
180
REGISTRAR LA ID DE NODO DEL NODO DE TRANSACCIÓN DE UNIÓN Y ACTUALIZAR LA LISTA LOCAL DE TODOS LOS NODOS CONECTADOS; CREAR UNA NUEVA CONEXIÓN
181
ENVIAR UN MENSAJE DE "BIENVENIDA" AL NODO DE TRANSACCIÓN DE UNIÓN CON LA IDENTIDAD DEL NODO DE ENVÍO
182
RECIBIR EL MENSAJE DE "BIENVENIDA"
183
ENVIAR UN MENSAJE DE PETICIÓN DE OBJETO AL NODO SELECCIONADO PARA UNA COPIA DE SU ÁTOMO DE CATÁLOGO MAESTRO
184
ACTUALIZAR Y SERIALIZAR EL ÁTOMO DE CATÁLOGO MAESTRO EN UN MENSAJE DE "OBJETO" Y DIFUNDIR EL MENSAJE DE "OBJETO" Y UN MENSAJE DE "OBJETO DISPONIBLE" A OTROS NODOS 173
185
RECUPERAR LOS ÁTOMOS GESTORES DE TRANSACCIÓN Y BASE DE DATOS EN RESPUESTA A MOTORES DE PETICIÓN DE BASE DE DATOS
186
CADA UNO DE LOS OTROS NODOS 173 ENVÍA UN MENSAJE DE "ACUSADO RECIBO DE OBJETO" AL NODO SELECCIONADO

FIG. 18

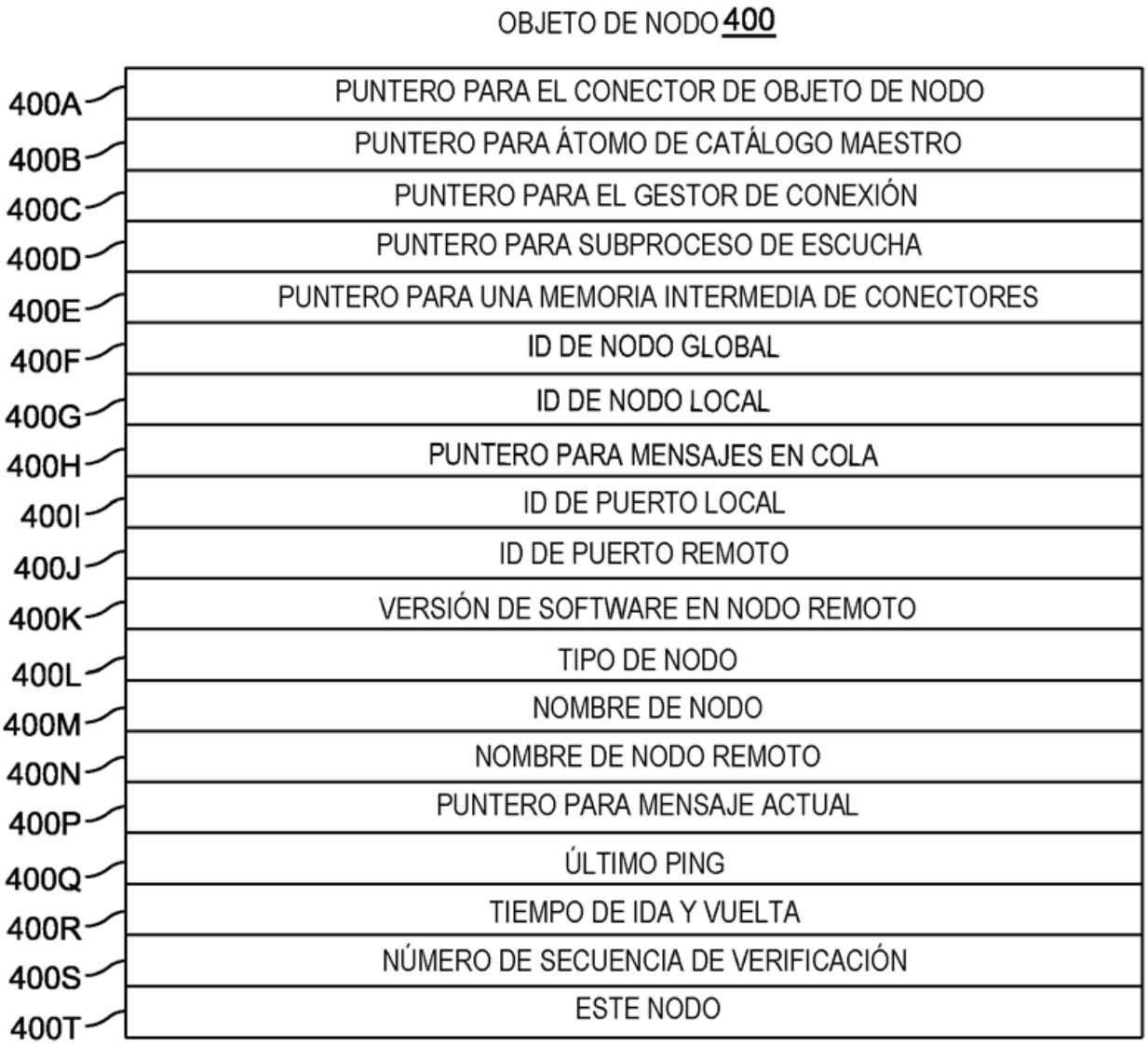
OBJETO DE NODO 400
400A
PUNTERO PARA EL CONECTOR DE OBJETO DE NODO
400B
PUNTERO PARA ÁTOMO DE CATÁLOGO MAESTRO
400C
PUNTERO PARA EL GESTOR DE CONEXIÓN
400D
PUNTERO PARA SUBPROCESO DE ESCUCHA
400E
PUNTERO PARA UNA MEMORIA INTERMEDIA DE CONECTORES
400F
ID DE NODO GLOBAL
400G
ID DE NODO LOCAL
400H
PUNTERO PARA MENSAJES EN COLA
400I
ID DE PUERTO LOCAL
400J
ID DE PUERTO REMOTO
400K
VERSIÓN DE SOFTWARE EN NODO REMOTO
400L
TIPO DE NODO
400M
NOMBRE DE NODO
400N
NOMBRE DE NODO REMOTO
400P
PUNTERO PARA MENSAJE ACTUAL
400Q
ÚLTIMO PING
400R
TIEMPO DE IDA Y VUELTA
400S
NÚMERO DE SECUENCIA DE VERIFICACIÓN
400T
ESTE NODO

FIG. 19

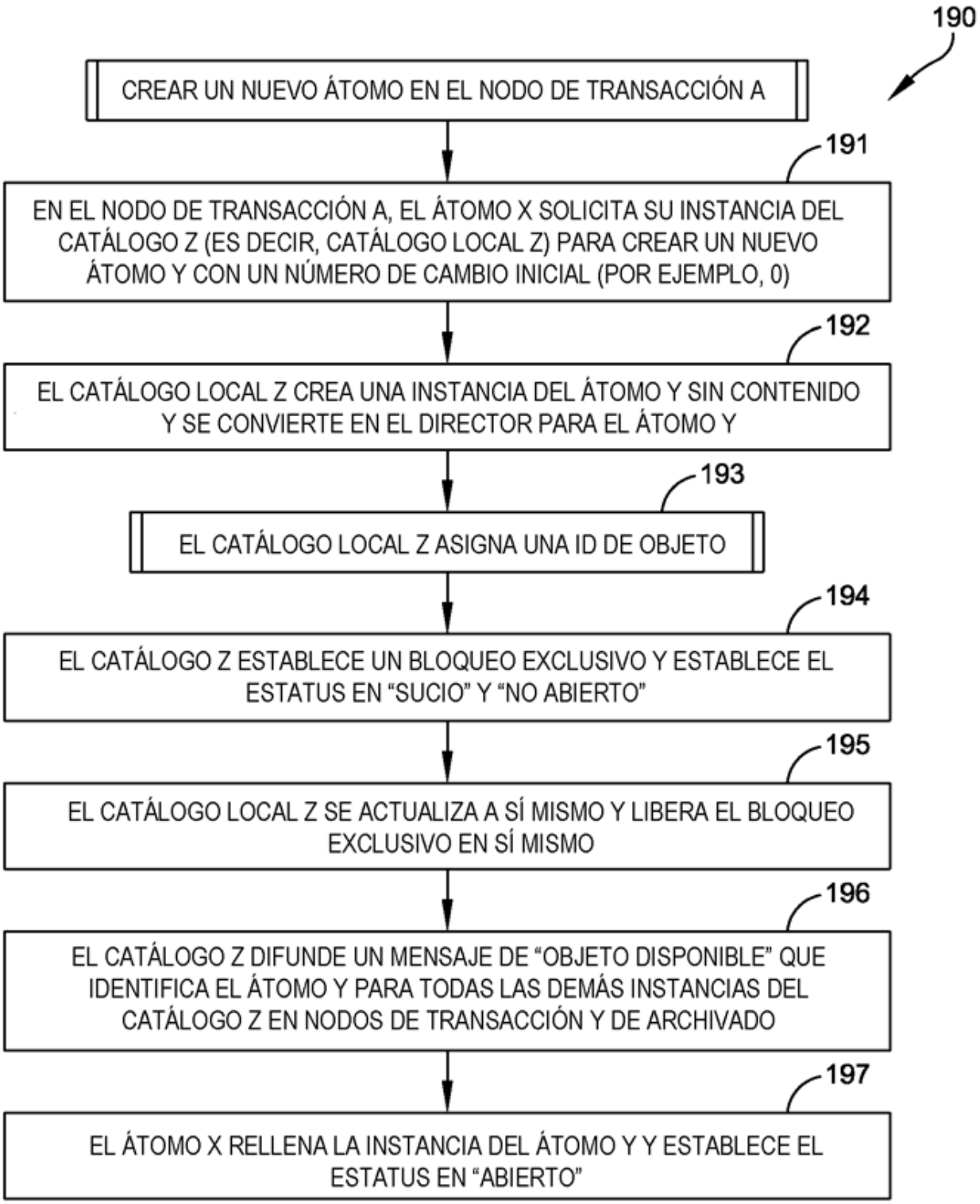
190
CREAR UN NUEVO ÁTOMO EN EL NODO DE TRANSACCIÓN A
191
EN EL NODO DE TRANSACCIÓN A, EL ÁTOMO X SOLICITA SU INSTANCIA DEL CATÁLOGO Z (ES DECIR, CATÁLOGO LOCAL Z) PARA CREAR UN NUEVO ÁTOMO Y CON UN NÚMERO DE CAMBIO INICIAL (POR EJEMPLO, 0)
192
EL CATÁLOGO LOCAL Z CREA UNA INSTANCIA DEL ÁTOMO Y SIN CONTENIDO Y SE CONVIERTE EN EL DIRECTOR PARA EL ÁTOMO Y
193
EL CATÁLOGO LOCAL Z ASIGNA UNA ID DE OBJETO
194
EL CATÁLOGO Z ESTABLECE UN BLOQUEO EXCLUSIVO Y ESTABLECE EL ESTATUS EN "SUCIO" Y "NO ABIERTO"
195
EL CATÁLOGO LOCAL Z SE ACTUALIZA A SÍ MISMO Y LIBERA EL BLOQUEO EXCLUSIVO EN SÍ MISMO
196
EL CATÁLOGO Z DIFUNDE UN MENSAJE DE "OBJETO DISPONIBLE" QUE IDENTIFICA EL ÁTOMO Y PARA TODAS LAS DEMÁS INSTANCIAS DEL CATÁLOGO Z EN NODOS DE TRANSACCIÓN Y DE ARCHIVADO
197
EL ÁTOMO X RELLENA LA INSTANCIA DEL ÁTOMO Y Y ESTABLECE EL ESTATUS EN "ABIERTO"

FIG. 20

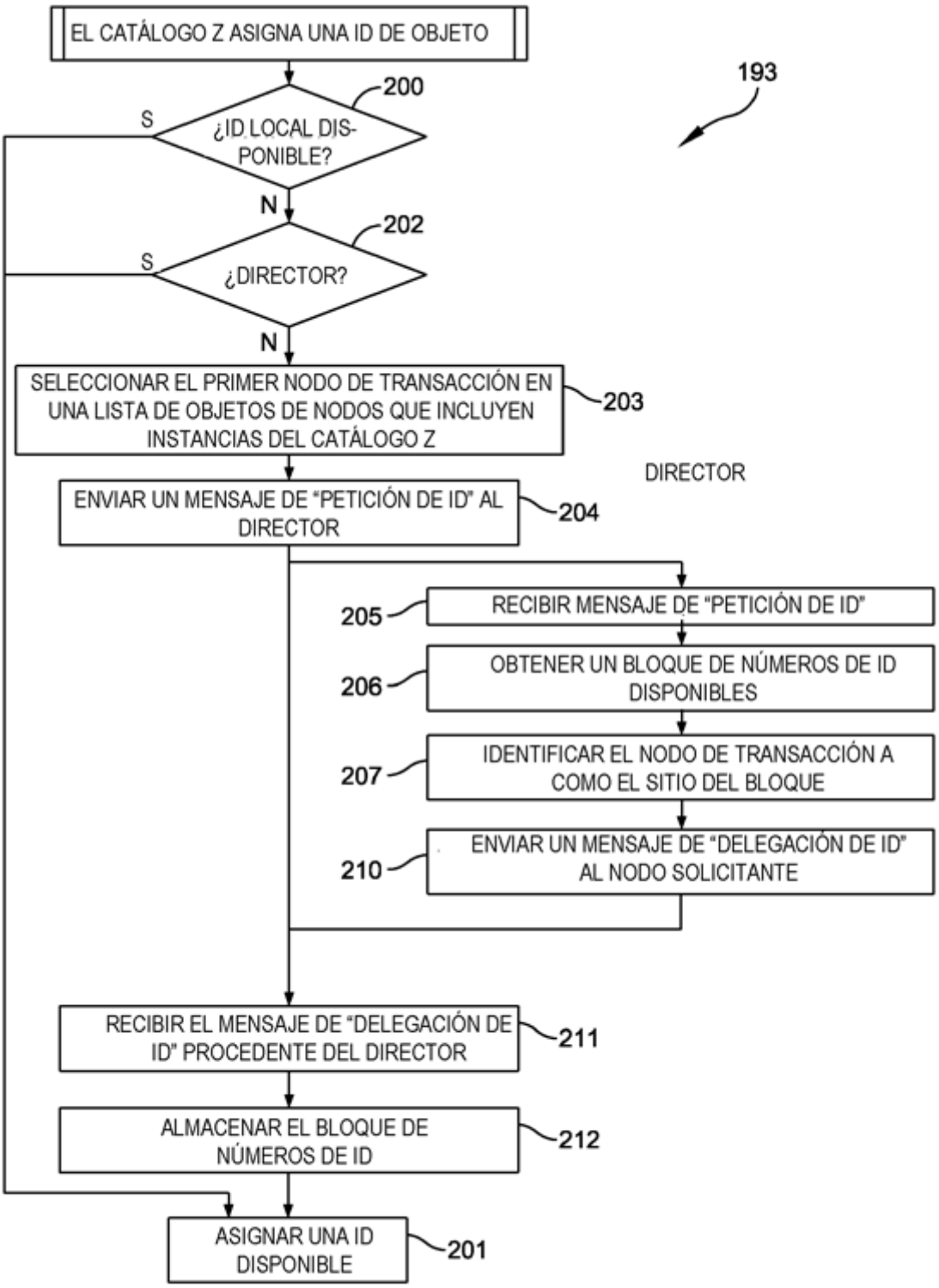
EL CATÁLOGO Z ASIGNA UNA ID DE OBJETO
193
200
S
¿ID LOCAL DISPONIBLE?
N
202
S
¿DIRECTOR?
N
SELECCIONAR EL PRIMER NODO DE TRANSACCIÓN EN UNA LISTA DE OBJETOS DE NODOS QUE INCLUYEN INSTANCIAS DEL CATÁLOGO Z
203
DIRECTOR
ENVIAR UN MENSAJE DE "PETICIÓN DE ID" AL DIRECTOR
204
205
RECIBIR MENSAJE DE "PETICIÓN DE ID"
206
OBTENER UN BLOQUE DE NÚMEROS DE ID DISPONIBLES
207
IDENTIFICAR EL NODO DE TRANSACCIÓN A COMO EL SITIO DEL BLOQUE
210
ENVIAR UN MENSAJE DE "DELEGACIÓN DE ID" AL NODO SOLICITANTE
RECIBIR EL MENSAJE DE "DELEGACIÓN DE ID" PROCEDENTE DEL DIRECTOR
211
ALMACENAR EL BLOQUE DE NÚMEROS DE ID
212
ASIGNAR UNA ID DISPONIBLE
201

FIG. 21

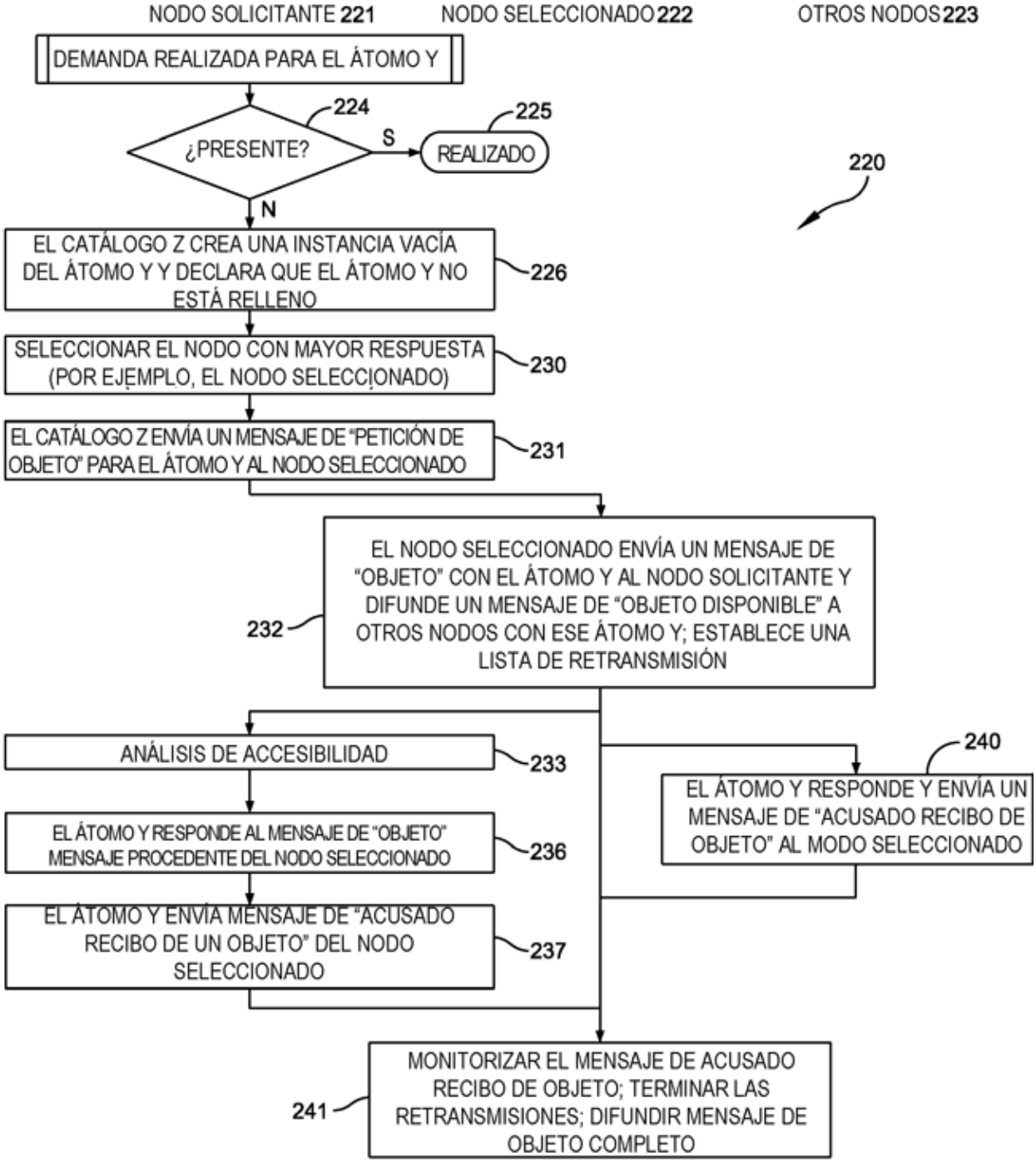
NODO SOLICITANTE 221
NODO SELECCIONADO 222
OTROS NODOS 223
DEMANDA REALIZADA PARA EL ÁTOMO Y
224
¿PRESENTE?
S
225
REALIZADO
N
220
EL CATÁLOGO Z CREA UNA INSTANCIA VACÍA DEL ÁTOMO Y Y DECLARA QUE EL ÁTOMO Y NO ESTÁ RELLENO
226
SELECCIONAR EL NODO CON MAYOR RESPUESTA (POR EJEMPLO, EL NODO SELECCIONADO)
230
EL CATÁLOGO Z ENVÍA UN MENSAJE DE "PETICIÓN DE OBJETO" PARA EL ÁTOMO Y AL NODO SELECCIONADO
231
EL NODO SELECCIONADO ENVÍA UN MENSAJE DE "OBJETO" CON EL ÁTOMO Y AL NODO SOLICITANTE Y DIFUNDE UN MENSAJE DE "OBJETO DISPONIBLE" A OTROS NODOS CON ESE ÁTOMO Y; ESTABLECE UNA LISTA DE RETRANSMISIÓN
232
ANÁLISIS DE ACCESIBILIDAD
233
240
EL ÁTOMO Y RESPONDE Y ENVÍA UN MENSAJE DE "ACUSADO RECIBO DE OBJETO" AL MODO SELECCIONADO
EL ÁTOMO Y RESPONDE AL MENSAJE DE "OBJETO" MENSAJE PROCEDENTE DEL NODO SELECCIONADO
236
EL ÁTOMO Y ENVÍA MENSAJE DE "ACUSADO RECIBO DE UN OBJETO" DEL NODO SELECCIONADO
237
MONITORIZAR EL MENSAJE DE ACUSADO RECIBO DE OBJETO; TERMINAR LAS RETRANSMISIONES; DIFUNDIR MENSAJE DE OBJETO COMPLETO
241

FIG. 22

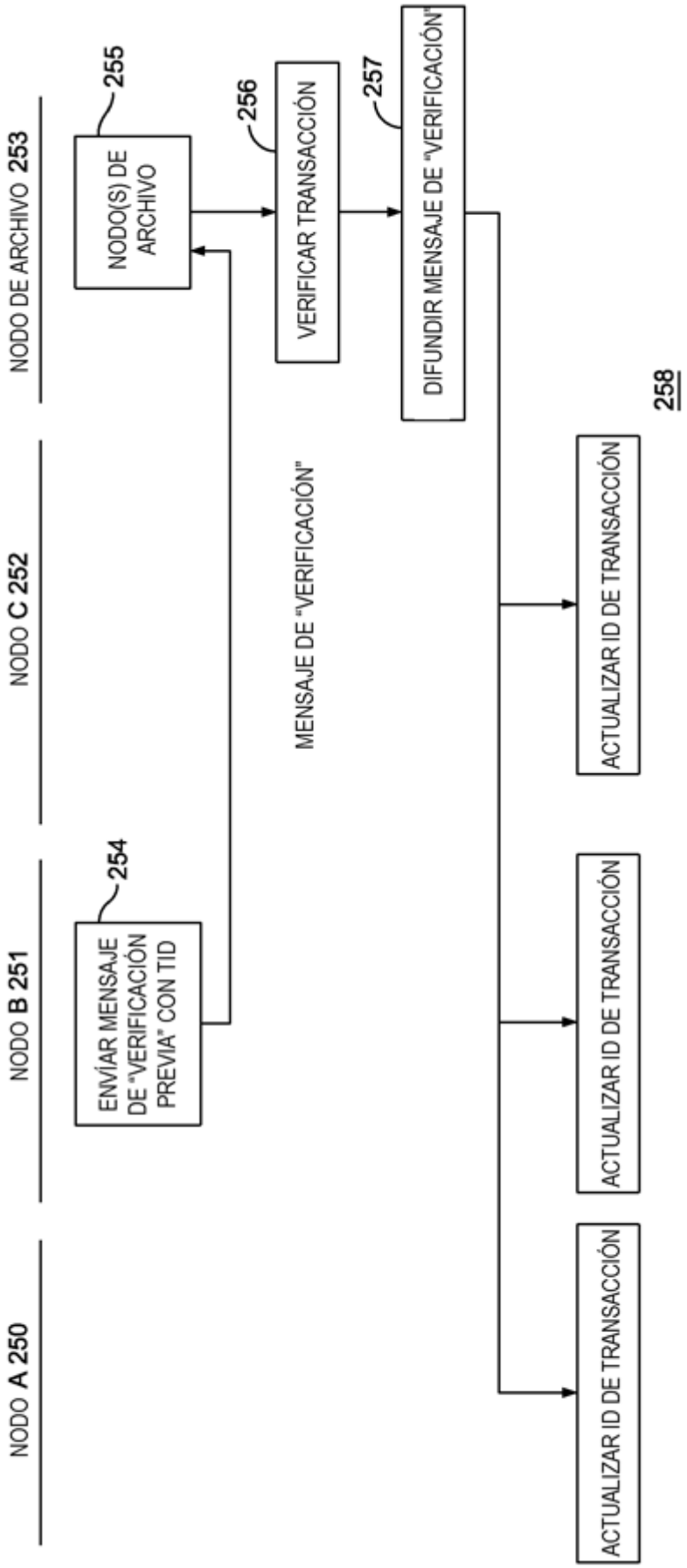
NODO A 250
NODO B 251
NODO C 252
NODO DE ARCHIVO 253
ENVÍAR MENSAJE DE "VERIFICACIÓN PREVIA" CON TID
254
NODO(S) DE ARCHIVO
255
VERIFICAR TRANSACCIÓN
256
DIFUNDIR MENSAJE DE "VERIFICACIÓN"
257
MENSAJE DE "VERIFICACIÓN"
ACTUALIZAR ID DE TRANSACCIÓN
ACTUALIZAR ID DE TRANSACCIÓN
ACTUALIZAR ID DE TRANSACCIÓN
258

FIG. 23