(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 691 254**

(51) Int. CI.:

G06F 21/51 (2013.01)
G06F 21/60 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **12.05.2017 PCT/IB2017/052800**

(87) Fecha y número de publicación internacional: **16.11.2017 WO17195160**

(96) Fecha de presentación y número de la solicitud europea: **12.05.2017 E 17724453 (0)**

(97) Fecha y número de publicación de la concesión europea: **11.07.2018 EP 3295349**

(54) Título: **Método y sistema para verificar la integridad de un activo digital mediante el uso de una tabla de hash distribuidas y un libro mayor distribuido entre pares**

(30) Prioridad:

13.05.2016 GB 201608463
13.05.2016 GB 201608456
13.05.2016 GB 201608454

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
26.11.2018

(73) Titular/es:

NCHAIN HOLDINGS LIMITED (100.0%)
Fitzgerald House 44 Church Street
St. John's, AG

(72) Inventor/es:

WRIGHT, CRAIG STEVEN y
SAVANAH, STEPHANE

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Método y sistema para verificar la integridad de un activo digital mediante el uso de una tabla de hash distribuidas y un libro mayor distribuido entre pares.

Campo técnico

La presente descripción se refiere a métodos de seguridad, control y verificación para asegurar y mantener la integridad de un activo digital. La invención es particularmente apropiada para verificar la propiedad y/o integridad de un artículo de software informático. Ello puede comprender el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares (cadena de bloques).

Antecedentes

En el presente documento, usamos el término "cadena de bloques" para incluir todas las formas de libros mayores electrónicos, distribuidos, basados en ordenador. Estos incluyen tecnologías de cadena de bloques basada en consenso y cadena de transacciones, libros mayores autorizados y no autorizados, libros mayores compartidos y sus variaciones. La aplicación más ampliamente conocida de la tecnología de cadena de bloques es la contabilidad Bitcoin, aunque se han propuesto y desarrollado otras implementaciones de cadena de bloques. Mientras, en la presente memoria, puede hacerse referencia a Bitcoin a los fines de conveniencia e ilustración, debe notarse que la invención no se encuentra limitada al uso con la cadena de bloques Bitcoin y las implementaciones y protocolos alternativos de cadena de bloques caen dentro del alcance de la presente invención. El término "usuario" puede hacer referencia, en la presente memoria, a un recurso humano o basado en procesador.

Una cadena de bloques es un libro mayor electrónico entre pares que se implementa como un sistema distribuido descentralizado basado en ordenador compuesto de bloques que, a su vez, se componen de transacciones. Cada transacción es una estructura de datos que codifica la transferencia de control de un activo digital entre participantes en el sistema de cadena de bloques, e incluye al menos una entrada y al menos una salida. Cada bloque contiene un *hash* del bloque previo de modo que los bloques se encadenan juntos para crear un registro permanente e inalterable de todas las transacciones que se han escrito en la cadena de bloques desde su comienzo. Las transacciones contienen pequeños programas conocidos como *scripts* incorporados en sus entradas y salidas, los cuales especifican cómo y quién puede acceder a las salidas de las transacciones. En la plataforma Bitcoin, dichos *scripts* se escriben mediante el uso de un lenguaje de *scripts* basado en una pila.

Para que una transacción se escriba en la cadena de bloques, debe "validarse". Los nodos de red (mineros) llevan a cabo trabajo para asegurar que cada transacción sea válida, con transacciones inválidas rechazadas desde la red. Los clientes de software instalados en los nodos llevan a cabo dicho trabajo de validación en una transacción no utilizada (UTXO) mediante la ejecución de sus *scripts* de bloqueo y desbloqueo. Si la ejecución de los *scripts* de bloqueo y desbloqueo es VERDADERA, la transacción es válida y la transacción se escribe en la cadena de bloques. Por consiguiente, para que una transacción se escriba en la cadena de bloques, debe i) validarse por el primer nodo que recibe la transacción -si la transacción se valida, el nodo la retransmite a los otros nodos en la red; y ii) añadirse a un bloque nuevo construido por un minero; y iii) minarse, a saber, añadirse al libro mayor público de transacciones pasadas.

Aunque la tecnología de cadena de bloques es más ampliamente conocida para el uso de la implementación de la criptomoneda, los emprendedores digitales han comenzado a explorar el uso tanto del sistema de seguridad criptográfico en el que se basa Bitcoin como los datos que pueden almacenarse en la Cadena de Bloques para implementar nuevos sistemas. Sería altamente ventajoso si la cadena de bloques pudiera usarse para tareas y procesos automatizados que no se encuentran limitados al ámbito de la criptomoneda. Dichas soluciones pueden emplear los beneficios de la cadena de bloques (p.ej., registros permanentes de prueba alterada de eventos, procesamiento distribuido, etc.) mientras son más versátiles en sus aplicaciones.

Un área de investigación actual es el uso de la cadena de bloques para la implementación de "contratos inteligentes". Estos son programas de ordenador diseñados para automatizar la ejecución de los términos de un contrato o acuerdo legible por máquina. A diferencia de un contrato tradicional que se escribe en lenguaje natural, un contrato inteligente es un programa ejecutable por máquina que comprende reglas que pueden procesar entradas con el fin de reducir resultados, lo cual puede entonces hacer que se lleven a cabo acciones según dichos resultados.

Otra área de interés relacionada con la cadena de bloques es el uso de "*tokens*" (o "monedas coloreadas") para representar y transferir entidades del mundo real mediante la cadena de bloques. Un artículo potencialmente sensible o secreto puede representarse por el *token* que no tiene un significado o valor discernible. El *token*, por consiguiente, sirve como un identificador que permite que se haga referencia al artículo del mundo real desde la cadena de bloques.

Debido al registro de prueba alterada que pueden proveer, las cadenas de bloques son apropiadas para aplicaciones en las cuales el control, la visibilidad de episodios y las transacciones/intercambios seguros son importantes. Una de dichas áreas de aplicación apropiadas es el intercambio o transferencia de activos digitales como, por ejemplo, software. Los enfoques tradicionales para asegurar la integridad y compartición de software informático implican la firma digital de los ejecutables del software informático. Por ejemplo, la firma del ejecutable u otro código asociado con un par criptográfico de claves como, por ejemplo, una clave pública y una clave privada. La clave pública se obtiene, con frecuencia, de una autoridad central fiable como, por ejemplo, una autoridad de certificación.

El software informático se acompaña, con frecuencia, de una licencia que contiene obligaciones contractuales. La licencia puede contener los términos que regulan el uso o la redistribución del software. Una cuestión puede surgir donde el software informático o la licencia asociada se transfieren, de manera ilícita, a otro usuario.

El software informático o programas de ordenador requieren, en general, una instalación antes de que la ejecución de las instrucciones allí contenidas pueda ocurrir. La instalación prepara el software informático o programa para la ejecución. El software informático o programas vienen, con frecuencia, con un instalador que lleva a cabo la instalación. Después de haber llevado a cabo la instalación una vez, esta no necesita llevarse a cabo nuevamente, y el software informático o programa pueden ejecutarse una y otra vez.

Es fundamental verificar la propiedad e integridad del software informático tras la instalación. Ello es para asegurar que, por ejemplo, el software informático no se haya transferido al propietario incorrecto y que el software informático no se haya corrompido o alterado durante el tránsito.

Cualquier descripción de documentos, actos, materiales, dispositivos, artículos o similares que se haya incluido en la presente memoria descriptiva no se tomará como una admisión de que cualquiera o todas de dichas cuestiones forman parte de la base de la técnica anterior o eran de conocimiento general común en el campo relevante a la presente descripción dado que existía antes de la fecha de prioridad de cada reivindicación de la presente solicitud.

A lo largo de la presente memoria descriptiva, se comprenderá que la palabra "comprender" o variaciones como, por ejemplo, "comprende" o "comprenden", implican la inclusión de un elemento establecido, entero o etapa, o grupo de elementos, enteros o etapas, pero no la exclusión de cualquier otro elemento, entero o etapa, o grupo de elementos, enteros o etapas.

El documento de Michael Crosby y otros: "*Blockchain Technology Beyond Bitcoin*", 16 oct 2015, XP055363520 resume todos los proyectos de cadena de bloques entonces en curso.

Compendio

La invención provee método(s) y sistema(s) correspondiente(s) según se define en las reivindicaciones anexas. La invención puede proveer un método/sistema de control y verificación implementado por ordenador. Este puede permitir o facilitar la transferencia de un activo controlado entre usuarios en una red basada en ordenador. El activo puede ser un activo digital. En la presente memoria, el término "usuario" puede usarse para hacer referencia a un recurso basado en ordenador. El activo controlado puede ser una porción o artículo de software. La invención puede proveer un método implementado por ordenador para verificar la propiedad y/o integridad de un activo controlado, p.ej., una porción de software informático. El software puede verificarse para la instalación en un recurso basado en ordenador. La verificación puede implicar permitir o habilitar a un usuario para usar o interactuar con el software o de otra manera llevar a cabo cierto acto en relación con el software si se ha establecido una concordancia.

El método puede comprender un método implementado por ordenador para verificar la integridad de un activo digital como, por ejemplo, una porción o artículo de software informático para la instalación, mediante el uso de una tabla de *hash* distribuidas (DHT, por sus siglas en inglés) y un libro mayor distribuido entre pares (cadena de bloques). Puede comprender determinar un metadato (M) asociado a un registro de transacciones (Tx) almacenado en el libro mayor distribuido entre pares; determinar una indicación de una entrada almacenada en la tabla de *hash* distribuidas del metadato (M); determinar un tercer valor *hash* (H3) basado en el software informático; determinar un cuarto valor *hash* (H4) de la entrada en la tabla de *hash* distribuidas; comparar el tercer valor *hash* (H3) y el cuarto valor *hash* (H4); y verificar la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4).

Por consiguiente, la invención incorpora el uso de fuentes técnicas separadas, a saber, una cadena de bloques y una DHT, respectivamente. Por consiguiente, la invención puede comprender el uso de recursos de almacenamiento técnicamente diferentes y distintos, con intercomunicación y transferencia de datos entre ellos. Mediante la búsqueda, procesamiento y recuperación de datos de una DHT y una cadena de bloques, la invención puede lograr los efectos de control, seguridad y verificación mejorados que resultan en un enfoque más seguro con respecto a la instalación de un activo digital (p.ej., software), verificación, transferencia y autorización. También provee un sistema informático mejorado dado que provee un mecanismo para asegurar la integridad de software (a saber, que el

software no se ha alterado en modo alguno con respecto a su estado original o previsto) y, por consiguiente, se ejecutará según lo esperado.

En el método, la comparación del tercer valor *hash* (H3) y cuarto valor *hash* (H4) puede comprender determinar si el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) concuerdan. El significado del término "concordar" puede comprender una correspondencia, igualdad o asociación entre los artículos comparados.

En el método, antes de determinar el metadato (M), el método puede comprender determinar una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) a partir de un registro de transacciones (Tx) almacenado en el libro mayor distribuido entre pares; determinar una segunda clave pública (P2) asociada al segundo usuario (U2) de una entrada almacenada en la tabla de *hash* distribuidas; comparar la clave pública del segundo usuario (PU2) y la segunda clave pública (P2); y verificar la propiedad del software informático según la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2). En el método, la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) puede comprender determinar si la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) concuerdan.

En el método, antes de determinar la clave pública de segundo usuario (PU2), el método puede comprender determinar un dato (D1) asociado al software informático; determinar un primer valor *hash* (H1) del software informático; determinar un segundo valor *hash* (H2) según el dato (D1) y el software informático; enviar, en una red de comunicaciones, el dato (D1), el primer valor *hash* (H1) y el segundo valor *hash* (H2) a una entrada para el almacenamiento en la tabla de *hash* distribuidas, en donde el segundo valor *hash* (H2) es una clave de un par de valores de claves y el dato (D1) y el primer valor *hash* (H1) son un valor en el par de valores de claves; y determinar el metadato (M) que comprende el segundo valor *hash* (H2) para el almacenamiento en el libro mayor distribuido entre pares.

En el método, el software informático puede comprender un encabezamiento y un cuerpo. El tercer valor *hash* (H3) puede determinarse a partir del cuerpo de software informático. El encabezamiento puede comprender un valor *hash* del cuerpo del software informático. El encabezamiento puede además comprender el segundo valor *hash* (H2). El cuerpo del software informático puede comprender un ejecutable del software informático.

En el método, antes de determinar la clave pública de segundo usuario (PU2), el método puede comprender encriptar el ejecutable del software informático. La encriptación del ejecutable del software informático puede comprender determinar un valor de generador (VG); determinar una segunda clave pública de segundo usuario (P2U2) según la clave pública de segundo usuario (PU2) y el valor de generador (VG), en donde la segunda clave pública de segundo usuario (P2U2) forma un par criptográfico con una segunda clave privada de segundo usuario (V2U2); determinar una segunda clave privada de primer usuario V2U1 según una clave privada de primer usuario VU1 y el valor de generador VG, en donde la clave privada de primer usuario VU1 forma un par criptográfico con una clave pública de primer usuario PU1; determinar un secreto común (SC) según la segunda clave pública de segundo usuario (P2U2) y la segunda clave privada de primer usuario (V2U1); y encriptar el software informático con el secreto común (SC) para generar un ejecutable encriptado del software informático.

En el método, el ejecutable encriptado del software informático puede desencriptarse mediante la determinación del secreto común (SC) según la segunda clave pública de primer usuario (P2U1) y la segunda clave privada de segundo usuario (V2U2); y mediante la desencriptación del ejecutable del software informático con el secreto común (SC) para generar un ejecutable desencriptado del software informático. También puede comprender: determinar una segunda clave privada de segundo usuario V2U2 según una clave privada de segundo usuario VU2 y el valor de generador VG, en donde la clave privada de segundo usuario VU2 forma un par criptográfico con una clave pública de segundo usuario PU2; y/o determinar una segunda clave privada de primer usuario V2U1 según una clave privada de primer usuario VU1 y el valor de generador VG, en donde la clave privada de primer usuario VU1 forma un par criptográfico con una clave pública de primer usuario PU1.

El método puede además comprender la instalación del ejecutable desencriptado del software informático en un dispositivo de procesamiento asociado al segundo usuario (U2).

El método puede además comprender la determinación de una clave de activación (AK, por sus siglas en inglés) del segundo usuario (U2); y la ejecución de instrucciones del ejecutable desencriptado del software informático según la clave de activación (AK).

Un programa de software informático que comprende instrucciones legibles por máquina para hacer que un dispositivo de procesamiento implemente el método descrito más arriba.

Un sistema informático para verificar la integridad de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares, el sistema comprendiendo un dispositivo de procesamiento asociado a un nodo en una red de nodos entre pares, configurado para determinar un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido entre pares; determinar una indicación de la ubicación de una entrada en la tabla de *hash* distribuidas del metadato (M); determinar un tercer

valor *hash* (H3) según el software informático; determinar un cuarto valor *hash* (H4) de la entrada en la tabla de *hash* distribuidas; comparar el tercer valor *hash* (H3) y el cuarto valor *hash* (H4); y verificar la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4).

Breve descripción de los dibujos

La Figura 1 ilustra un ejemplo de una tabla de *hash*.

Ejemplos de la presente descripción se describirán con referencia a:

La Figura 2 ilustra un diagrama esquemático de un sistema a modo de ejemplo para determinar un metadato (M) para asegurar el software informático de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas;

la Figura 3 ilustra un diagrama de flujo de un método implementado por ordenador para determinar un metadato (M) para asegurar un software informático mediante el uso de una tabla de *hash* distribuidas;

la Figura 4 ilustra un ejemplo de un árbol de Merkle;

la Figura 5 ilustra un ejemplo de un árbol de Merkle con referencia a un software informático y a una licencia asociada a un software informático;

la Figura 6 ilustra un diagrama de flujo de un método implementado por ordenador para determinar un identificador indicativo de la ubicación de un software informático mediante el uso de una tabla de *hash* distribuidas;

la Figura 7 ilustra un diagrama de flujo de un método implementado por ordenador para verificar la propiedad de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares;

la Figura 8 ilustra un diagrama de flujo de métodos implementados por ordenador para determinar un secreto común;

la Figura 9 ilustra un diagrama de flujo de métodos implementados por ordenador para encriptar un ejecutable de un software informático;

la Figura 10 ilustra un diagrama de flujo de un método implementado por ordenador para verificar la integridad de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares; y

la Figura 11 ilustra un esquema de un dispositivo de procesamiento a modo de ejemplo.

Descripción de las realizaciones

La presente descripción se refiere, en general, a métodos y sistemas para utilizar una tabla de *hash* distribuidas y un libro mayor distribuido entre pares (P2P, por sus siglas en inglés) como, por ejemplo, la cadena de bloques Bitcoin, para permitir la verificación de un software informático para la instalación.

Mientras las realizaciones descritas más abajo pueden referirse específicamente a transacciones que ocurren en la cadena de bloques Bitcoin (a la que, en la presente memoria, se hace referencia como la cadena de bloques), se apreciará que la presente invención puede implementarse mediante el uso de otros libros mayores distribuidos P2P. La cadena de bloques se usa más abajo para describir aspectos de la invención en aras de la simplicidad solamente debido a su alto nivel de normalización y gran cantidad de documentación pública asociada.

Tabla de *hash* distribuidas

En un modelo de cliente/servidor típico, un servidor central puede estar a cargo de la mayoría de recursos. Ello significa que, en el caso de un ataque o fallo en el servidor central, la mayoría de los recursos almacenados en el servidor central pueden verse comprometidos. Por el contrario, en un modelo distribuido, los recursos se comparten ("distribuyen") entre nodos participantes. De esta manera, la capacidad de todos los nodos participantes se utiliza y el fallo de un servidor no comprometerá a la mayoría de los recursos.

La Figura 1 ilustra un ejemplo de una tabla de *hash*. La tabla de *hash* consta de pares de valores de claves. La clave de cada par de valores de claves se mapea, a modo de una función *hash*, a un índice. El índice define la ubicación de valores almacenados de los pares de valores de claves.

Una DHT es un ejemplo de aplicación del modelo distribuido a una tabla de *hash*. De manera similar a una tabla de *hash*, una DHT comprende pares de valores de claves y provee un método eficaz para ubicar ("consultar") un valor de un par de valores de claves dada simplemente la clave. Sin embargo, a diferencia de la tabla de *hash*, los pares

de valores de claves se distribuyen y almacenan por un número de nodos participantes. De esta manera, la responsabilidad de almacenar y mantener los pares de valores de claves se comparte por los nodos participantes.

En la misma manera que una tabla de *hash*, cada par de valores de claves en la DHT se mapea a un índice. El índice se determina para cada par de valores de claves llevando a cabo una función *hash* en la clave. Por ejemplo, el Algoritmo de *Hash* Seguro SHA-1 criptográfico puede usarse para determinar el índice.

A cada nodo participante se le asigna al menos un índice mediante la partición de espacio de claves. Para cada índice al que el nodo participante se asigna, el nodo participante almacena el valor de dicho par de valores de claves.

Es una ventaja que valores de los pares de valores de claves pueden recuperarse de manera eficaz. Con el fin de recuperar un valor asociado a una clave, un nodo puede ejecutar una "consulta" para determinar el nodo responsable (mediante el índice). Puede entonces accederse al nodo responsable para determinar el valor.

Bitcoin y la cadena de bloques

Como se conoce en la técnica, la cadena de bloques es un libro mayor de tipo de transacción de base de datos donde la capacidad de almacenamiento se distribuye a lo largo de nodos interconectados que participan en un sistema basado en el protocolo Bitcoin. Cada transacción Bitcoin se radiodifunde a la red, las transacciones se confirman y luego se agregan a bloques. Los bloques se incluyen entonces en la cadena de bloques mediante el almacenamiento de los bloques en múltiples nodos participantes.

Una copia total de un libro mayor distribuido P2P de criptomoneda contiene cada transacción ejecutada en la criptomoneda. Por consiguiente, se provee una lista continuamente creciente de registros de datos transaccionales. Dado que cada transacción ingresada en la cadena de bloques se ejecuta de manera criptográfica, la cadena de bloques se endurece contra la alteración y revisión, incluso por operadores de los nodos participantes.

Debido a la transparencia de la cadena de bloques, hay historias públicamente disponibles para cada transacción.

Es una ventaja adicional de la cadena de bloques que la transacción y el registro de la transacción sean iguales.

De esta manera, la información relacionada con la transacción se captura en la transacción real. Dicho registro es permanente e inmutable y, por lo tanto, elimina el requisito de que un tercero mantenga el registro de la transacción en una base de datos separada.

Pago a *hash* de *script* y multifirma

Mientras las realizaciones de más abajo pueden referirse específicamente a transacciones que usan el método de pago a un *hash* de *script* (P2SH, por sus siglas en inglés) del protocolo Bitcoin, se apreciará que la presente invención puede implementarse mediante el uso de otro método del protocolo Bitcoin como, por ejemplo, el método de pago a *hash* de clave pública.

Cada registro de transacción en la cadena de bloques comprende un *script* que incluye información indicativa de la transacción y un número de claves públicas. Dichas claves públicas pueden asociarse al emisor y receptor de la criptomoneda. Un *script* puede considerarse una lista de instrucciones registradas en cada registro de transacción en la cadena de bloques que describe cómo un usuario puede obtener acceso a la criptomoneda especificada en el registro de transacción.

Como antecedente, en un método P2SH estándar del protocolo Bitcoin, el *script* de salida, o *script* de rescate, pueden tomar la forma:

<NumSigs PubK1 PubK2 ... PubK15 NumKeys OP_CHECKMULTISIG>

donde NumSigs es el número "m" de firmas válidas requeridas para satisfacer el *script* de rescate para desbloquear la transacción; PubK1, PubK2 ... PubK15 son las claves públicas que corresponden a firmas que desbloquean la transacción (hasta un máximo de 15 claves públicas) y NumKeys es el número "n" de claves públicas.

En el protocolo Bitcoin, las firmas basadas en la clave privada de un usuario pueden generarse mediante el uso del Algoritmo de Firma Digital con Curva Elíptica. Las firmas se usan entonces para el rescate de la criptomoneda asociada al *script* de salida o *script* de rescate. Cuando un usuario rescata un *script* de salida o *script* de rescate, el usuario provee su firma y clave pública. El *script* de salida o *script* de rescate luego verifica la firma contra la clave pública.

Con el fin de rescatar el *script* de rescate de más arriba, se requiere al menos un número "m" de firmas correspondientes a las claves públicas. En algunos ejemplos, el orden de las claves públicas es importante y el número "m" de "n" firmas para la firma debe llevarse a cabo en secuencia. Por ejemplo, es preciso considerar donde

"m" es 2 y "n" es 15. Si hay dos firmas disponibles para su uso, Sig1 (correspondiente a PubK1) y Sig15 (correspondiente a PubK15), el *script* de rescate debe firmarse por Sig1 primero seguida de Sig15.

Resumen del sistema

Ahora se describirán un método, dispositivo y sistema para determinar un metadato (M) para asegurar un software informático y verificar la propiedad de un software informático para la instalación.

La Figura 2 ilustra un sistema 1 que incluye un primer nodo 3 que está en comunicación con, en una red de comunicaciones 5, un segundo nodo 7. El primer nodo 3 tiene un primer dispositivo de procesamiento 21 asociado y el segundo nodo 5 tiene un segundo dispositivo de procesamiento 27 asociado. Ejemplos del primer y segundo nodos 3, 7 incluyen un dispositivo electrónico como, por ejemplo, un ordenador, tableta, dispositivo de comunicaciones móviles, servidor de ordenador, etc.

Una DHT 13 para registrar y almacenar pares de valores de claves también se ilustra en la Figura 2. La DHT 13 puede asociarse a uno o más dispositivos de procesamiento 19 para recibir, registrar y almacenar los valores de los pares de valores de claves. Los dispositivos de procesamiento 19 pueden usarse por nodos participantes de la DHT 13. Según se describe más arriba, la DHT 13 provee un método eficaz para ubicar valores de pares de valores de claves.

La Figura 2 también ilustra un libro mayor distribuido P2P 14 para registrar transacciones. El libro mayor distribuido P2P 14 puede asociarse a uno o más dispositivos de procesamiento 20 para recibir y registrar transacciones. Según se describe más arriba, un ejemplo de un libro mayor distribuido P2P 14 es la cadena de bloques Bitcoin. Por lo tanto, en el contexto de la cadena de bloques, los dispositivos de procesamiento 20 asociados al libro mayor distribuido P2P 14 pueden ser dispositivos de procesamiento a los que se hace referencia como "mineros".

El primer nodo 3 se asocia a un primer usuario 23 y el segundo nodo 7 se asocia a un segundo usuario 24. En un ejemplo, el primer nodo 3 puede representar un fabricante del software informático. En otro ejemplo, el primer nodo 3 puede representar un agente o proveedor de servicios. En incluso otro ejemplo, el primer nodo 3 puede representar un usuario del software informático.

El segundo nodo 7 puede representar un usuario del sistema informático. En otro ejemplo, el segundo nodo 7 puede representar un agente, proveedor de servicios o fabricante del software informático.

En un ejemplo, el primer nodo 3 lleva a cabo el método 100, 300, 400, 500, 600, 700, 800 según se ilustra por la Figura 3, Figura 6, Figura 7, Figura 8 y Figura 9. En otro ejemplo, el segundo nodo 7 lleva a cabo el método 100, 300, 400, 500, 600, 700, 800.

Mientras las realizaciones a modo de ejemplo de más abajo pueden referirse al primer nodo 3 como uno que lleva a cabo los métodos o al segundo nodo 7 como uno que lleva a cabo los métodos, se comprenderá que la descripción también puede adaptarse o modificarse para llevarse a cabo por otros nodos.

El método 100 según se ilustra por la Figura 3 asegura el software informático e incluye determinar 110 un dato (D1) asociado al software informático. El dato (D1) puede además comprender una licencia asociada al software informático. El método 100 también incluye determinar 120 un primer valor *hash* (H1) según el software informático. En un ejemplo, el primer valor *hash* (H1) puede relacionarse con un ejecutable del software informático.

El método 100 también incluye determinar 130 un segundo valor *hash* (H2) según el dato (D1) y el software informático. En un ejemplo, el segundo valor *hash* (H2) puede ser representativo de los detalles del software informático y de la licencia asociada al software informático. En un ejemplo adicional, el segundo valor *hash* (H2) puede comprender información adicional.

El método 100 además incluye enviar 140, en una red de comunicaciones 5, el dato (D1), el primer valor *hash* (H1) y el segundo valor *hash* (H2) a una entrada en una DHT 13, en donde el segundo valor *hash* (H2) se asigna a una clave de un par de valores de claves y el dato (D1) y el primer valor *hash* (H1) se asignan al valor en el par de valores de claves. El valor en el par de valores de claves puede además comprender un identificador indicativo de la ubicación del software informático o licencia.

El método 100 también incluye determinar 150 un metadato (M) que se basa en el segundo valor *hash* (H2) para su inclusión en el libro mayor distribuido entre pares 14. En un ejemplo, el metadato (M) puede incluirse en un primer *script* de rescate (RS1) para su inclusión en el libro mayor distribuido entre pares 14.

El método 600 según se ilustra por la Figura 7 verifica la propiedad del software informático y se lleva a cabo después del método descrito más arriba. Ello se muestra como la etapa 100 opcional en la Figura 7. El método 600 incluye determinar 610 una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) de un registro de transacciones almacenado en el libro mayor distribuido entre pares 14. La clave pública de segundo usuario (PU2) puede incluirse en un *script* de salida del registro de transacciones. En otro ejemplo, la clave pública

de segundo usuario (PU2) puede incluirse en el metadato (M) que se encuentra en el libro mayor distribuido entre pares 14 según se describe más arriba.

El método 600 también incluye determinar 620 una segunda clave pública (P2) asociada al segundo usuario (U2) de una entrada almacenada en la DHT 13. La segunda clave pública (P2) puede ser igual a la clave pública de segundo usuario (PU2). La entrada en la DHT 13 puede comprender un par de valores de claves.

El método 600 además incluye comparar 630 la clave pública de segundo usuario (PU2) y la segunda clave pública (P2). El método 600 también incluye verificar 640 la propiedad del software informático según la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2). En un ejemplo, la verificación de la propiedad puede indicar que la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) concuerdan. Es decir, una concordancia entre la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) puede indicar que la propiedad se ha verificado.

El método 900 según se ilustra por la Figura 10 verifica la integridad del software informático y se lleva a cabo después del método descrito más arriba. Ello se muestra como la etapa 600 opcional en la Figura 10. El método 900 incluye determinar 910 un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido entre pares 14. El metadato (M) puede incluirse en un *script* de salida del registro de transacciones. El método 900 también incluye determinar 920 una indicación de una entrada almacenada en la DHT 13 del metadato (M). En un ejemplo, una indicación de una entrada puede comprender una dirección que identifica una entrada en la tabla de *hash* distribuidas 13.

El método 900 también incluye determinar 930 un tercer valor *hash* (H3) según el software informático. En un ejemplo, el tercer valor *hash* (H3) se calcula según los contenidos del software informático. El método también incluye determinar 640 un cuarto valor *hash* (H4) de la entrada en la DHT 13.

El método 900 además incluye comparar 950 el tercer valor *hash* (H3) y el cuarto valor *hash* (H4). El método 900 también incluye verificar 960 la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4). En un ejemplo, la verificación de la integridad puede indicar que el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) concuerdan. Es decir, una concordancia entre el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) puede indicar que la integridad se ha verificado.

Ahora se describirá un ejemplo detallado del método 100, 600, 900.

Determinación de un dato asociado al software informático 110

Según se describe más arriba, el método 100 incluye determinar 110 un dato (D1) asociado al software informático. La determinación 110 de un dato (D1) puede comprender recibir el dato (D1) de un usuario, nodo o almacén de datos. La determinación 110 de un dato (D1) puede además comprender generar el dato (D1) en el primer nodo 3.

En un ejemplo, el primer nodo 3 puede recibir el dato (D1) del primer usuario 23 mediante la interfaz de usuario 15. En otro ejemplo, el primer nodo 3 puede recibir el dato (D1) del segundo usuario 24. En incluso otro ejemplo, el primer nodo 3 puede recibir el dato (D1) de un almacén de datos 17.

El dato (D1) se asocia al software informático donde el dato (D1) puede identificar el software informático, información adicional, una licencia del software informático o ser indicativo de la ubicación del software informático. Por ejemplo, el dato (D1) puede comprender una cadena o estructura de datos que identifica el software informático. La cadena o estructura de datos puede comprender una recopilación de palabras clave de identificación y/o información adicional sobre el software informático. Un ejemplo de información adicional puede ser un identificador de la versión del software informático, por ejemplo, un numeral. Por ejemplo, si el software informático se titula BobSoftware y la versión es 3.0, la cadena o estructura de datos (D1) puede comprender "BobSoftware/3.0".

En un ejemplo adicional, el dato (D1) puede comprender un identificador de una licencia asociada al software informático. Este puede ser un número de identificación (ID) de licencia de software o una clave de licencia de software. En otro ejemplo, el identificador de la licencia puede comprender un *hash* criptográfico de los contenidos de la licencia.

El dato (D1) puede además comprender un identificador indicativo de la ubicación de almacenamiento del software informático. En un ejemplo, el identificador puede comprender un URL para un objeto en Internet. En un ejemplo adicional, puede proveerse un enlace a la ubicación de almacenamiento del software informático en un depósito como, por ejemplo, una tabla de *hash* o tabla de *hash* distribuidas.

En incluso un ejemplo adicional, el dato (D1) puede comprender información que identifica al fabricante del software informático. Ello puede incluir detalles personales como, por ejemplo, nombre, dirección, detalles de contacto o una clave pública asociada al fabricante.

Determinación de un primer valor *hash* (H1) según el software informático 120

Según se describe también más arriba, el método 100 además incluye determinar 120 un primer valor *hash* (H1) del software informático. La determinación 120 de un primer valor *hash* (H1) puede comprender recibir el primer valor *hash* (H1) de un usuario o acceder al primer valor *hash* (H1) desde un almacén de datos. La determinación 120 de un primer valor *hash* (H1) puede además comprender calcular el valor *hash* en el primer nodo 3.

En un ejemplo, el primer nodo 3 puede recibir el primer valor *hash* (H1) del primer usuario 23 mediante la interfaz de usuario 15. En otro ejemplo, el primer nodo 3 puede recibir el primer valor *hash* (H1) del segundo usuario 24. En incluso otro ejemplo, el primer nodo 3 puede acceder al primer valor *hash* (H1) desde un almacén de datos local 17 o almacén de datos remoto.

En un ejemplo, el primer valor *hash* (H1) es de un ejecutable del software informático. El ejecutable del software informático puede recuperarse de la red de comunicaciones 5 como, por ejemplo, Internet. En otro ejemplo, el ejecutable puede proveerse por el primer usuario 23 o segundo usuario 24. En incluso otro ejemplo, el ejecutable puede recuperarse del almacén de datos 17. En incluso un ejemplo adicional, el ejecutable puede recuperarse de un depósito como, por ejemplo, una tabla de *hash* o una DHT.

El *hash* del ejecutable del software puede determinarse mediante el uso del algoritmo SHA-256 para crear una representación de 256 bits de la información. Se apreciará que otros algoritmos de *hash* pueden usarse, incluidos otros algoritmos en la familia Algoritmo de *Hash* Seguro (SHA). Algunos ejemplos particulares incluyen instancias en el subconjunto SHA-3, incluidos SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256. Otros algoritmos de *hash* pueden incluir aquellos en la familia Condensado de Mensaje de Evaluación de Primitivos de Integridad RACE (RIPEMD, por sus siglas en inglés). Un ejemplo particular puede incluir RIPEMD-160. Otras funciones *hash* pueden incluir familias basadas en la función *hash* Zémor-Tillich y funciones *hash* basadas en mochila.

Determinación de un segundo valor *hash* (H2) según el dato (D1) y el software informático 130

El método 100 también incluye determinar 130 un segundo valor *hash* (H2) según el dato (D1) y el software informático.

En un ejemplo, el segundo valor *hash* (H2) puede determinarse según el *hash* de la concatenación del dato (D1) y el ejecutable (o *hash* del ejecutable, es decir, el primer valor *hash* (H1)) del software informático. En un ejemplo adicional, el segundo valor *hash* (H2) puede determinarse según el *hash* de la concatenación del dato (D1), el ejecutable (o *hash* del ejecutable) del software informático e información adicional.

La información adicional puede comprender una clave pública del primer usuario 23 (PU1) o segundo usuario 24 (PU2). En un ejemplo adicional, la información adicional puede comprender un identificador de una entidad asociada al primer usuario 23 o segundo usuario 24. Por ejemplo, la entidad puede ser un empleador del primer usuario 23 o segundo usuario 24. En otro ejemplo, la entidad puede ser un proveedor de servicios del primer usuario 23 o segundo usuario 24.

La información adicional puede además comprender un identificador de dispositivo de un dispositivo asociado al primer nodo 3, segundo nodo 7, primer usuario 23 o segundo usuario 24. Un ejemplo de un dispositivo es el primer dispositivo de procesamiento 21 según se ilustra en la Figura 2. El identificador de dispositivo puede comprender al menos uno de los siguientes: una dirección MAC, número de serie de placa madre o un número de identificación de dispositivo. El identificador de dispositivo puede además ser una concatenación de al menos dos de la dirección MAC, número de serie de placa madre o número de identificación de dispositivo. En un ejemplo adicional, el identificador de dispositivo puede comprender un valor *hash* asociado a la dirección MAC, número de serie de placa madre o número de identificación de dispositivo, o la concatenación descrita más arriba.

En incluso un ejemplo adicional, la información adicional puede comprender una fecha de expiración de la licencia asociada al software informático.

Licencia asociada al software informático

En un ejemplo adicional, el segundo valor *hash* (H2) puede determinarse según la concatenación del dato (D1), el ejecutable (o *hash* del ejecutable) del software informático, información adicional o la licencia que se relaciona con el software informático.

La representación de la licencia puede ser un archivo o documento que especifica el contenido de la licencia. Por ejemplo, un texto claro ASCII, documento PDF o documento Word. El segundo valor *hash* (H2) puede incluir la licencia en su forma original o, por ejemplo, puede proveer un enlace a la ubicación de la licencia en una red de comunicaciones públicamente accesible como, por ejemplo, Internet. En un ejemplo adicional, puede proveerse un enlace a la ubicación de la licencia en un depósito como, por ejemplo, una tabla *hash* o DHT. En incluso un ejemplo adicional, puede proveerse un enlace a la ubicación de la licencia en un recurso basado en ordenador como, por ejemplo, el almacén de datos 17.

En un ejemplo, la licencia puede comprender el primer valor *hash* (H1) asociado al software informático.

La licencia asociada al software informático puede además comprender información adicional según se describe más arriba. En un ejemplo, la licencia puede asociarse a un primer usuario 23 o segundo usuario 24. La licencia puede comprender la clave pública del primer usuario 23 (PU1) o segundo usuario 24 (PU2). En un ejemplo adicional, la licencia puede comprender un identificador de una entidad asociada al primer usuario 23 o segundo usuario 24.

La licencia asociada al software informático puede además comprender un identificador de dispositivo de un dispositivo asociado al primer nodo 3, segundo nodo 7, primer usuario 23 o segundo usuario 24. Un ejemplo de un dispositivo es el primer dispositivo de procesamiento 21 según se ilustra en la Figura 2. El identificador de dispositivo puede comprender al menos uno de los siguientes: una dirección MAC, número de serie de placa madre o un número de identificación de dispositivo. El identificador de dispositivo puede además ser una concatenación de al menos dos de la dirección MAC, número de serie de placa madre o número de identificación de dispositivo. En un ejemplo adicional, el identificador de dispositivo puede comprender un valor *hash* asociado a la dirección MAC, número de serie de placa madre o número de identificación de dispositivo, o la concatenación descrita más arriba.

El primer usuario 23 puede ser el fabricante del software informático y el segundo usuario 24 puede ser el receptor ("usuario final") del software informático. En otro ejemplo, el segundo usuario 24 puede ser el fabricante del software informático y el primer usuario 23 puede ser el usuario final del software informático.

En un ejemplo, la licencia asociada al software informático puede autorizar solamente a un usuario final (una "licencia de un solo usuario"). En un ejemplo adicional, la licencia asociada al software informático puede autorizar un dispositivo del usuario final (una "licencia de un solo dispositivo"). En otro ejemplo, la licencia asociada al software informático puede autorizar más de un dispositivo del usuario final (una "licencia multidispositivo").

En otro ejemplo, puede haber más de un usuario final (una "licencia multiusuario"). En un ejemplo adicional, la licencia asociada al software informático puede autorizar un dispositivo por usuario final. En otro ejemplo, la licencia asociada al software informático puede autorizar más de un dispositivo por usuario final.

En el caso en el que la licencia se asocia a un primer usuario 23 o segundo usuario 24, la licencia puede comprender la clave pública de primer usuario (PU1) asociada al primer usuario 23 y la clave pública de segundo usuario (PU2) asociada al segundo usuario 24.

Árbol de Merkle

En otro ejemplo, la licencia puede ser el valor *hash* superior de un árbol de Merkle. Un ejemplo de un árbol de Merkle se ilustra en la Figura 4. En un árbol de Merkle, el valor *hash* en cada nodo son *hashes* de sus respectivos nodos "hijos". Por ejemplo, el valor *hash* Hash-A 305 es el *hash* de los valores *hash* en los dos nodos "hijos" 309 y 311. Puede verse que el valor *hash* superior del árbol de Merkle, Hash-AB 303, comprende todos los valores *hash* en el árbol de Merkle. Es decir, captura los valores *hash* de las cuatro "hojas" en la parte inferior del árbol, A1 317, A2 319, B1 321 y B2 323.

En un ejemplo de la presente descripción, cada "hoja" del árbol de Merkle puede representar un aspecto de la información de la licencia. Una licencia a modo de ejemplo se ilustra en la Figura 5. El dato (D1) 417 se captura en el valor *hash* Hash-D 409, el ejecutable del software 419 se captura en el valor *hash* Hash-S 411 (H1), las claves públicas 421 de usuarios 23 y/o 24 se capturan en el valor *hash* Hash-P 413 y la fecha de expiración 423 se captura en el valor *hash* Hash-E 415. Puede verse que los nodos 405 y 407 capturan los valores *hash* asociados a las hojas para el dato (D1) 417 y software 419, y claves públicas 421 y fecha de expiración 423 respectivamente.

Se apreciará que otra información no descrita de otra manera más arriba puede comprender la información adicional en la que se basa el valor *hash* (H2).

Envío del dato (D1), primer valor *hash* (H1) y segundo valor *hash* (H2) a una tabla de *hash* distribuidas 140

El método 100 también incluye enviar 140, en una red de comunicaciones 5, el dato (D1), primer valor *hash* (H1) y el segundo valor *hash* (H2) a una entrada en una tabla de *hash* distribuidas 13.

En un ejemplo, el segundo valor *hash* (H2) puede ser una clave de un par de valores de claves, y el dato (D1) y el primer valor *hash* (H1) pueden ser un valor en el par de valores de claves.

En un ejemplo adicional, información adicional según se describe más arriba puede también ser parte del valor en el par de valores de claves. Ello incluye, pero sin limitación: claves públicas del primer usuario 23 o segundo usuario 24, un identificador de dispositivo de un dispositivo asociado al primer nodo 3, segundo nodo 7, primer usuario 23 o segundo usuario 24, un identificador indicativo de la ubicación del software informático o licencia, o información adicional asociada a la licencia.

Según se describe más arriba, una DHT 13 consta de pares de valores de claves, donde cada par de valores de claves se asigna a un índice. En un ejemplo, el segundo valor *hash* (H2) puede usarse para generar el índice. Una función *hash* o función *hash* criptográfica puede llevarse a cabo en el segundo valor *hash* (H2). Por ejemplo, la función criptográfica SHA-1 puede usarse:

$$\text{Índice} = \text{SHA-1}(\text{H2})$$

Para que el segundo valor *hash* (H2) sea la clave de un par de valores de claves en la DHT 13, y el dato (D1) y el primer valor *hash* (H1) sean el valor en el par de valores de claves, la clave y valor se envían a cualquier nodo participante de la DHT 13.

En un ejemplo, un mensaje como, por ejemplo, *put*(*key, value*) puede enviarse a un nodo participante de la DHT 13, donde *key* es el segundo valor *hash* (H2) y *value* es el dato (D1) y el primer valor *hash* (H1). El mensaje puede enviarse a todos los nodos participantes hasta que se recibe por el nodo participante que se asigna al índice según se indica por la participación del espacio de claves. El nodo participante asignado al índice indicado en el mensaje puede entonces almacenar el par de valores de claves en la DHT 13 y asumir la responsabilidad de mantener la entrada asociada al par de valores de claves.

Es una ventaja que el valor de cualquier clave dada puede recuperarse de la DHT 13. En un ejemplo, el primer usuario 23 o segundo usuario 24 pueden desear recuperar el valor. El primer usuario 23 o segundo usuario 24, mediante el primer nodo 3, segundo nodo 7 u otro nodo no ilustrado de otra forma, pueden proveer a cualquier nodo participante de la DHT 13 un mensaje de solicitud como, por ejemplo, *get(key).* El mensaje de solicitud puede entonces enviarse a todos los nodos participantes hasta que se recibe por el nodo participante que se asigna al índice según se indica por la participación del espacio de claves.

Determinación de un metadato (M) 150

El método 100 además incluye determinar 150 un metadato (M) que comprende el segundo valor *hash* (H2). La determinación 150 de un metadato (M) puede comprender recibir el metadato (M) de un usuario, nodo o almacén de datos. El metadato (M) puede incluirse, por ejemplo, en uno o más de los 15 lugares disponibles para las claves públicas en un primer *script* de rescate (RS1) multifirma P2SH de una transacción en el libro mayor distribuido P2P 14.

El primer *script* de rescate (RS1) de la transacción en el libro mayor distribuido P2P 14 puede representar una emisión, o creación, de una transacción *tokenizada* ("*token* de emisión") que representa el contenido incluido en el metadato (M). En un ejemplo, el *token* puede emitirse por un agente (A).

En el método P2SH del protocolo Bitcoin, el metadato puede incluirse en un *script* de rescate por medio del proceso provisto más abajo.

Metadatos

El metadato (M) puede incorporarse en uno o más de los 15 lugares disponibles para las claves públicas en un *script* de rescate (RS1) multifirma P2SH. Por ejemplo, el *script* de rescate (RS1) puede tomar la forma de:

<NumSigs Metadata1 Metadata2... PubK1 PubK2... NumKeys OP_CHECKMULTISIG>

donde Metadata1 y Metadata2 incluyen, cada uno, un metadato que toma el lugar de una clave pública en el *script* de rescate y PubK1 y PubK2 son claves públicas.

El metadato (M) puede comprender el segundo valor *hash* (H2). El metadato (M) puede además comprender una descripción o palabra clave que describe condiciones asociadas al software informático o licencia. Por ejemplo, la fecha de la licencia, nombre, fecha de nacimiento, dirección, detalles de contacto, u otros detalles del usuario asociado a la licencia. En un ejemplo adicional, la información asociada a la cantidad de criptomoneda puede incluirse.

El metadato (M) puede incluir la información en una cantidad de maneras. En un ejemplo, los contenidos de la información pueden incluirse. En un ejemplo adicional, un *hash* criptográfico de la información puede incluirse. El *hash* de la información puede determinarse mediante el uso del algoritmo SHA-256 para crear una representación de 256 bits de la información. Se apreciará que otros algoritmos de *hash* pueden usarse, incluidos otros algoritmos en la familia Algoritmo de *Hash* Seguro (SHA). Algunos ejemplos particulares incluyen instancias en el subconjunto SHA-3, incluidos SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256. Otros algoritmos de *hash* pueden incluir aquellos en la familia Condensado de Mensaje de Evaluación de Primitivos de Integridad RACE (RIPEMD). Un ejemplo particular puede incluir RIPEMD-160. Otras funciones *hash* pueden incluir familias basadas en la función *hash* Zémor-Tillich y funciones *hash* basadas en mochila.

En realizaciones adicionales de la presente descripción, combinaciones que incluyen uno o más de lo descrito más arriba pueden incluirse en el metadato (M). Dado que el metadato (M) puede hacerse público por medio del libro mayor distribuido P2P 14 como, por ejemplo, la cadena de bloques, o transmitirse en una red no segura, puede ser deseable que detalles específicos del metadato (M) se escondan u oculten por motivos de privacidad.

Por lo tanto, el uso de transacciones Bitcoin P2SH multifirma en realizaciones de la presente descripción ofrece una ventaja dado que permite la transferencia y el registro permanente de información asociada al software informático y a la licencia. Dicho registro se logra mediante la inclusión del metadato en el *script* de salida de una transacción, por ejemplo, un *script* de rescate.

Primer *script* de rescate

Según se describe más arriba, un *script* de rescate es un ejemplo de un *script* de salida en el método P2SH estándar del protocolo Bitcoin y describe cómo un usuario puede obtener acceso a la criptomoneda especificada en el registro de transacción.

En la presente descripción, el primer *script* de rescate (RS1) para el *token* de emisión puede basarse en el metadato (M). El primer *script* de rescate (RS1) puede además comprender una clave pública de agente (PA) que forma un par criptográfico con una clave privada de agente (VA). De esta manera, se requiere que la clave privada de agente (VA) "desbloquee" o utilice criptomoneda que se asocia a la transacción.

En un ejemplo, el primer *script* de rescate (RS1) para el *token* de emisión puede incluir el metadato (M). El primer *script* de rescate (RS1) puede además comprender una clave pública de agente (PA). En el presente ejemplo, el primer *script* de rescate (RS1) puede ser de la forma:

<OP_1 PA Metadata1 Metadata2 OP_3 OP_CHECKMULTISIG>

donde OP_1 denota el número de firmas requeridas para satisfacer el primer *script* de rescate (RS1) para desbloquear la transacción ("NumSigs"), y OP_3 denota el número de claves públicas en el *script* de rescate ("NumKeys").

En el presente ejemplo, el primer *script* de rescate (RS1) puede comprender dos campos designados para los metadatos, Metadata1 y Metadata2. Un ejemplo específico de los Metadata1 y Metadata2 se ilustra en la Tabla 1 más abajo.

Campo	Subcampo	Bytes	Comentarios
Metadata1	LicenceType	4	El valor codificado indica el tipo de licencia.
	LicencePointer	16	Dirección IPv6 que identifica la DHT.
	LicenceTypeData1	12	El formato depende del valor de LicenceType. Rellenado con ceros.
Metadata2	LicenceHash	20	RIPEMD-160(SHA256(archivo de licencia real direccionado por LicencePointer))
	LicenceTypeData2	12	El formato depende del valor de LicenceType. Rellenado con ceros.

Tabla 1

El presente ejemplo incluye proveer un puntero a la licencia en Metadata1 que puede ser útil donde el tamaño de la licencia impide incluir dichos detalles en el metadato (M). Además, dado que el metadato (M) puede hacerse público, o transmitirse en una red no segura, puede ser deseable que detalles específicos del *token* se escondan u oculten por motivos de privacidad.

Los primeros 4 bytes de Metadata1 indican el tipo de licencia. Por ejemplo, el tipo de licencia puede denotar el nombre del software informático como, por ejemplo, BobSoftware. En un ejemplo adicional, el tipo de licencia puede denotar el tipo de autorización de la licencia como, por ejemplo, "un solo usuario" o "multidispositivo", según se describe más arriba. Los siguientes 16 bytes mantienen la dirección IP de la ubicación del archivo de licencia electrónico real, y dejan un margen para las direcciones IPv6. Es preciso notar que, en algunas realizaciones, dicho valor puede apuntar a la semilla de un archivo *torrent* de modo que el archivo de licencia puede distribuirse en la nube antes que centralizarse. Los siguientes 12 bytes contienen datos específicos al tipo de licencia.

Los primeros 20 bytes de Metadata2 son un *hash* del archivo de licencia real mediante el uso de RIPEMD-160 en SHA256 aplicado a los contenidos reales del archivo de licencia. Dado que el archivo de licencia real puede ser recuperable, ello permite la validación de la transacción contra el contrato. Es preciso notar que el propio archivo de licencia puede ser completamente público (no encriptado y legible para el ser humano) o puede encriptarse en aras de la privacidad, según los requisitos de la realización específica. El contenido de los 12 bytes restantes de Metadata2 puede usarse según el tipo de licencia.

Puede verse a partir del ejemplo del primer *script* de rescate (RS1) provisto más arriba que el *token* de emisión debe firmarse por el agente (A) para poder enviarse. Un ejemplo de la transacción para el *token* de emisión se provee en la Tabla 2, donde, en aras de la brevedad, las tasas del minero no se muestran.

ID-600	Transacción- ID
Version number	Número de versión
1	Número de entradas
ID-110	Salida de Trans Prev
IDX-00	Índice de Salida de Trans Prev
Script length	Longitud de *script*
OP_0 Sig-VA < redeem script ID-110 >	ScriptSig
0000 0000 0000 0001	Número de secuencia
1	Número de salidas
C1	Valor de salida
Output script length	Longitud de *script* de salida
OP_HASH160 < hash of redeem script (RS1) > OP_EQUAL	*Script* de salida
LockTime	Tiempo de Bloqueo

Tabla 2

Las líneas 4 a 8 de la Tabla 2 representan la entrada en la transacción que es la primera cantidad de criptomoneda (C1) que se incluirá en el *token* de emisión (a saber, "tokenizado"). En el presente ejemplo, la primera cantidad de criptomoneda (C1) es el resultado de una transacción previa (ID-110) que transfiere la primera cantidad de criptomoneda en beneficio del agente (A) y, por lo tanto, el *script* de salida (*script* de rescate ID-110) de la transacción previa (ID-110) incluye la clave pública del agente (PA). Por consiguiente, para desbloquear dicha salida previa, el *script* (*script* de rescate ID-110) debe firmarse con la clave privada (VA) del primer usuario. Finalmente, la línea 8 de la Tabla 2 indica que la primera cantidad de criptomoneda (C1) será la primera salida en la presente transacción (ID-600).

Las líneas 9 a 13 de la Tabla 2 representan la primera (y única) salida de la transacción (ID-600) que, en el presente caso, es representativa del *token* de emisión creado y transferido otra vez al agente. La línea 10 muestra el valor de salida, que es la primera cantidad de criptomoneda (C1). La línea 11 muestra el *script* de salida, que incluye "< hash of redeem script >" según su uso en el método P2SH del protocolo Bitcoin. En el presente ejemplo, el *script* de rescate es el primer *script* de rescate (RS1) en la forma que se describe más arriba.

La salida de la transacción (ID-600) que se muestra en la Tabla 2 se registra entonces, con la primera salida de datos (O1), en el libro mayor distribuido P2P 14. En particular, la primera salida de datos (O1) puede comprender una indicación de la primera cantidad de criptomoneda (C1) que se transfiere en la transacción. La primera salida de datos (O1) puede además comprender un *hash* del primer *script* de rescate (RS1).

En transacciones futuras de la primera cantidad de criptomoneda (C1), por ejemplo, la transferencia del *token* a un primer usuario 23 o segundo usuario 24, el *script* para desbloquear la primera cantidad de criptomoneda (C1) (p.ej., la entrada ScriptSig de la transacción futura) puede ser en la forma:

OP_0 Sig-VA Sig-VU1 <OP_1 PA PU1 Metadata1 Metadata2 OP_4 OP_CHECKMULTISIG>

donde Sig-VU1 indica la firma del primer usuario 23. Es preciso notar que el *script* de más arriba supone que solo una firma del agente (A) o del primer usuario 23 se requiere para desbloquear la primera cantidad de criptomoneda (C1).

El *token* de emisión puede transferirse a otro usuario por medio de un segundo *script* de rescate (RS2).

Variaciones

Segundo *script* de rescate

El *token* que se asocia al software informático y a la licencia puede transferirse del agente (A) a otro usuario, por ejemplo, al primer usuario 23 o segundo usuario 24. En un ejemplo, la transferencia del *token* puede ser representativa de la autorización de acceso al usuario para el software informático o licencia. La transferencia puede implementarse por un segundo *script* de rescate (RS2).

En un ejemplo, el agente (A) desea transferir el *token* de emisión a un primer usuario 23. El primer usuario 23 puede representar, por ejemplo, a un fabricante del software informático.

En el presente ejemplo, el segundo *script* de rescate (RS2) puede basarse en el metadato (M), la clave pública de agente (PA) asociada al agente (A) y la clave pública de primer usuario (PU1) asociada al primer usuario 23.

El segundo *script* de rescate (RS2) puede ser de la forma:

<OP_1 PA_PU1 Metadata1 Metadata2 OP_4 OP_CHECKMULTISIG>

En el presente ejemplo, el segundo *script* de rescate (RS2) comprende los mismos dos campos de metadato que el primer *script* de rescate (RS1). El segundo *script* de rescate (RS2) además comprende la clave pública de agente (PA) asociada al agente y la clave pública de primer usuario (PU1) asociada al primer usuario.

Puede verse a partir del ejemplo del segundo *script* de rescate (RS2) provisto más arriba que el *token* que se transfiere debe firmarse por el agente (A) o primer usuario 23 para poder enviarse. Un ejemplo de la transacción para dicha transferencia del *token* de emisión se provee en la Tabla 3, donde, nuevamente, en aras de la brevedad, las tasas del minero no se muestran.

ID-610	Transacción-ID
Version number	Número de versión
1	Número de entradas
ID-600	Salida de Trans Prev
IDX-00	Índice de Salida de Trans Prev
Script length	Longitud de *script*
Sig-VA < OP_1 PA Metadata1 Metadata2 OP_3 OP_CHECKMULTISIG >	ScriptSig
0000 0000 0000 0001	Número de secuencia
1	Número de salidas
C1	Valor de salida
Output script length	Longitud de *script* de salida

OP _HASH160 < hash of redeem script (RS2) > OP_EQUAL	*Script* de salida
LockTime	Tiempo de Bloqueo

Tabla 3

De manera similar a la Tabla 2, las líneas 4 a 8 de la Tabla 3 representan la entrada en la transacción (ID-610). En el presente ejemplo, la entrada es el *token* de emisión, a saber, la salida de la transacción (ID-600) que se ilustra en la Tabla 2. Puede verse que el *script* de rescate en la línea 7 corresponde al *script* de rescate del *token* de emisión, a saber, el primer *script* de rescate (RS1). Por consiguiente, para desbloquear la salida de la transacción (ID-600), el primer *script* de rescate (RS1) debe firmarse con la clave pública del agente (PA).

Las líneas 9 a 13 de la Tabla 3 representan la salida de la transacción (ID-610) que, en el presente caso, es representativa del *token* de emisión que se transfiere al agente (A) o al primer usuario 23 (U1). La línea 10 muestra el valor de salida, que es la primera cantidad de criptomoneda (C1). La línea 11 muestra el *script* de salida, que incluye "< hash of redeem script >" según su uso en el método P2SH del protocolo Bitcoin. En el presente ejemplo, el *script* de rescate es el segundo *script* de rescate (RS2) en la forma que se describe más arriba.

La salida de la transacción (ID-610) se registra entonces, con una segunda salida de datos (O2), en el libro mayor distribuido P2P 14. La segunda salida de datos (O2) puede comprender una indicación de que la primera cantidad de criptomoneda (C1) de la primera salida de datos (O1) se transferirá en la transacción. La segunda salida de datos (O2) puede además comprender un *hash* del segundo *script* de rescate (RS2).

Identificador indicativo de la ubicación del software informático o licencia

Según se describe más arriba, el dato (D1) o licencia pueden comprender un identificador indicativo de la ubicación del software informático o licencia, respectivamente.

En un ejemplo, el identificador puede determinarse de forma independiente del dato (D1) o la licencia y permanecer separado del dato (D1) o licencia. El identificador puede además asignarse al valor del par de valores de claves junto con el dato (D1) y el primer valor *hash* (H1) según se describe en el método 100 de más arriba. De esta manera, el identificador puede incluirse en el campo *value* del mensaje *put*(*key, value*) y enviarse a un nodo participante en la DHT 13, según se describe más arriba.

En un ejemplo, el identificador indicativo de la ubicación puede comprender un URL para un objeto en Internet. En otro ejemplo, el identificador indicativo de la ubicación puede comprender una dirección para un depósito como, por ejemplo, una tabla de *hash* o una DHT 13. En incluso otro ejemplo, el identificador indicativo de la ubicación puede comprender una dirección para un depósito basado en ordenador como, por ejemplo, un servidor, base de datos o instalación de almacenamiento provistos en un recurso basado en ordenador como, por ejemplo, el almacén de datos 17 asociado al primer dispositivo de procesamiento 21 del primer nodo 3.

La Figura 6 ilustra un método 500 para determinar la ubicación del software informático o licencia. El método 500 incluye determinar 510 el metadato (M) a partir del primer *script* de rescate (RS1). Según se describe más arriba, el metadato (M) puede incorporarse en uno o más de los 15 lugares disponibles para las claves públicas en el primer *script* de rescate (RS1).

En el método P2SH del protocolo Bitcoin, cuando la salida de una transacción se utiliza en una transacción subsiguiente, el *script* de rescate se convierte en visible en la transacción subsiguiente. Según se describe más arriba y con referencia a la Tabla 2, la transacción (ID-600) para el *token* de emisión se devuelve al agente (A). De esta manera, el agente (A) puede utilizar dicho *token* de emisión para exponer el primer *script* de rescate (RS1). El metadato (M) que se basa en el segundo valor *hash* (H2) es, por lo tanto, visible en el libro mayor distribuido P2P 14. De esta manera, el segundo valor *hash* (H2) puede recuperarse 520 del metadato (M) en el primer *script* de rescate (RS1). En un ejemplo, el valor asociado a la clave del par de valores de claves puede recuperarse de la DHT 13 mediante el uso del mensaje de solicitud *get(key).*

El método 500 además incluye enviar 530, en una red de comunicaciones 5, el segundo valor *hash* (H2) a un procesador asociado a un nodo participante de la DHT 13. Según se describe más arriba, el segundo valor *hash* (H2) puede ser la clave del par de valores de claves. Según se describe también más arriba, el valor para una clave dada puede recuperarse mediante la provisión de un mensaje que contiene la clave a cualquier nodo participante de la DHT 13. Por lo tanto, en el ejemplo donde el identificador se incluye en el campo valor del par de valores de claves, el método 500 puede determinar 540, a partir del procesador del nodo participante, el identificador indicativo de la ubicación del software informático o licencia.

Determinación de una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) 610

Según se describe más arriba, el método 600 incluye determinar 610 una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) a partir de un registro de transacciones almacenado en el libro mayor distribuido P2P 14. La determinación de una clave pública de segundo usuario (PU2) a partir de un registro de transacciones puede comprender recibir el registro de transacciones de un usuario, nodo o almacén de datos y consultar el registro de transacciones para la clave pública de segundo usuario (PU2). La determinación de una clave pública de segundo usuario (PU2) a partir de un registro de transacciones puede además comprender acceder al registro de transacciones en un usuario, nodo o almacén de datos y consultar el registro de transacciones para la clave pública de segundo usuario (PU2).

En un ejemplo, el segundo nodo 7 asociado al segundo usuario 24 puede recibir el registro de transacciones del primer nodo 3 o de un almacén de datos 17 asociado al primer nodo 3. En otro ejemplo, el segundo nodo 7 puede recibir el registro de transacciones del primer usuario 23 o segundo usuario 24.

En incluso otro ejemplo, el segundo nodo 7 puede acceder al registro de transacciones en el segundo nodo 7 o en un almacén de datos asociado al segundo nodo 7. En un ejemplo adicional, puede accederse al registro de transacciones por el segundo nodo 7 mediante el uso de una instalación públicamente disponible como, por ejemplo, www.blockchain.info.

El registro de transacciones almacenado en el libro mayor distribuido P2P 14 puede comprender información que identifica la transacción o usuarios asociados a la transacción. Un ejemplo de la información comprendida en un registro de transacciones se muestra en la Tabla 4.

Campo	Descripción
Version number	Indica qué reglas del protocolo Bitcoin seguirá la transacción
Number of inputs	Número de entradas
Inputs	Al menos una entrada
Number of outputs	Número de salidas
Outputs	Al menos una salida
LockTime	Un sello temporal

Tabla 4

Cada salida de transacción incluye información sobre la cantidad de criptomoneda transferida y un *script* de salida que define las condiciones que se requiere que se satisfagan para utilizar la criptomoneda. El *script* de salida normalmente incluye una clave pública asociada a un receptor de la criptomoneda.

En un ejemplo, la clave pública asociada al receptor de la criptomoneda en el *script* de salida puede ser la clave pública de segundo usuario (PU2). De esta manera, la clave pública de segundo usuario (PU2) asociada al segundo usuario (U2) se determina a partir del *script* de salida en el registro de transacciones almacenado en el libro mayor distribuido P2P 14.

Según se describe más arriba, en el método P2SH del protocolo Bitcoin, el *script* de salida es el *script* de rescate. El *script* de rescate puede incluir un número de claves públicas asociadas al emisor y receptor de la criptomoneda. En un ejemplo, la clave pública de segundo usuario (PU2) asociada al segundo usuario (U2) puede determinarse a partir del *script* de rescate del registro de transacciones.

En otro ejemplo, la clave pública de segundo usuario (PU2) puede almacenarse en el metadato (M) del *script* de rescate. Según se describe más arriba, en el método P2SH, cuando la salida de la transacción se utiliza en una transacción subsiguiente, el *script* de rescate se convierte en visible en el libro mayor distribuido P2P 14. De esta manera, la clave pública de segundo usuario (PU2) puede recuperarse del metadato (M) en el *script* de rescate.

Determinación de una segunda clave pública (P2) asociada al segundo usuario (U2) 620

El método 600 además incluye determinar 620 una segunda clave pública (P2) asociada al segundo usuario (U2) a partir de una entrada almacenada en la DHT 13. La determinación de una segunda clave pública (P2) puede comprender recuperar un valor del par de valores de claves asociado a la entrada almacenada en la DHT 13. La

determinación de una segunda clave pública (P2) puede también comprender recibir el valor del par de valores de claves de otro nodo.

En un ejemplo, el valor del par de valores de claves asociado a la entrada en la DHT 13 puede recuperarse mediante el envío de un mensaje de solicitud a un nodo participante de la DHT 13. Según se describe más arriba, el mensaje de solicitud puede comprender *get(key),* donde *key* es la clave para el par de valores de claves asociado a la entrada en la DHT 13.

En un ejemplo adicional, la clave del par de valores de claves es el segundo valor *hash* (H2).

En otro ejemplo, el segundo nodo 7 puede recibir el valor almacenado en la DHT 13 del primer nodo 3 u otro nodo no ilustrado de otra manera. El primer nodo 3 u otro nodo pueden proveer a un nodo participante de la DHT 13 el mensaje de solicitud *get(key).* El primer nodo 3 u otro nodo pueden entonces recibir el valor del par de valores de claves asociado a la entrada en la DHT 13. El valor del par de valores de claves puede entonces enviarse al segundo nodo 7 desde el primer nodo 3 u otro nodo en la red de comunicaciones 5.

Comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) 630

El método además incluye comparar 630 la clave pública de segundo usuario (PU2) y la segunda clave pública (P2). La comparación puede comprender determinar si la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) concuerdan.

En un ejemplo, una concordancia puede indicar que la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) son equivalentes.

En otro ejemplo, una concordancia puede indicar que la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) pertenecen a la misma cartera de criptomoneda.

En un ejemplo adicional, la cartera de criptomoneda puede ser una cartera determinista y una concordancia puede indicar que la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) derivan de una semilla común. La semilla común puede ser una secuencia de caracteres.

Verificación de la propiedad del software informático según la comparación 640

El método 600 además incluye verificar 640 la propiedad del software informático según la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2).

En un ejemplo, la verificación de la propiedad del software informático ocurre si la comparación determina que la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) concuerdan.

Variaciones

Software informático

El software informático puede comprender un encabezamiento y un cuerpo. En un ejemplo, el encabezamiento puede comprender información asociada al software informático. En un ejemplo adicional, el encabezamiento puede comprender un valor *hash* del cuerpo del software informático. En incluso un ejemplo adicional, el encabezamiento puede comprender el segundo valor *hash* (H2) según se describe más arriba.

El cuerpo del software informático puede comprender un ejecutable del software informático.

Encriptación del ejecutable del software informático

En el método 600 descrito más arriba, antes de determinar la clave pública de segundo usuario (PU2), el método 600 puede comprender encriptar el ejecutable del software informático.

En un ejemplo, el ejecutable del software informático se encripta con una clave pública asociada al primer usuario 23 o segundo usuario 24. En otro ejemplo, el ejecutable del software informático se encripta con una clave pública asociada al primer nodo 3 o segundo nodo 7. En incluso otro ejemplo, el ejecutable del software informático se encripta con una clave pública asociada a un tercero o a un nodo no ilustrado de otra manera.

En otro ejemplo, el ejecutable del software informático puede encriptarse mediante el uso de un enfoque de compartición de secreto común similar a la técnica provista más abajo.

Determinación del secreto común (SC)

Un secreto común para la encriptación puede determinarse en los nodos 3, 7 por los usuarios 23, 24 asociados a los nodos llevando a cabo las etapas de los métodos 300, 400 respectivamente, según se ilustra en la Figura 8. De esta

manera, el secreto común puede determinarse de forma independiente sin comunicar claves privadas asociadas a los usuarios 23, 24 en la red de comunicaciones 5.

Según se ilustra en la Figura 8, el método 300 llevado a cabo por el primer usuario 23 incluye determinar 300 una segunda clave privada de primer usuario (V2U1) según al menos una clave privada de primer usuario (VU1) y un valor de generador (VG). La clave privada de primer usuario (VU1) forma un par criptográfico con la clave pública de primer usuario (PU1).

El valor de generador puede basarse en un mensaje que se comparte entre el primer usuario 23 y el segundo usuario 24, que puede incluir compartir el mensaje en la red de comunicaciones 5. El método 300 también incluye determinar 370 una segunda clave pública de segundo usuario (P2U2) según al menos la clave pública de segundo usuario (PU2) y el valor de generador (VG). El método 300 además incluye determinar 380, en el primer usuario 23, el secreto común (SC) según la segunda clave pública de segundo usuario (P2U2) y la segunda clave privada de primer usuario (V2U1).

De manera significativa, el mismo secreto común (SC) puede determinarse por el segundo usuario 24 asociado al segundo nodo 7 por el método 400. El método 400 incluye determinar 430 una segunda clave pública de primer usuario (P2U1) según la clave pública de primer usuario (PU1) y el valor de generador (VG). El método 400 además incluye determinar 470 una segunda clave privada de segundo usuario (V2U2) según la clave privada de segundo usuario (VU2) y el valor de generador (VG). La clave privada de segundo usuario (VU2) forma un par criptográfico con la clave pública de segundo usuario (PU2).

El método 400 además incluye determinar 480, en el segundo usuario 24, el secreto común (SC) según la segunda clave pública de primer usuario (P2U1) y la segunda clave privada de segundo usuario (V2U2). Los métodos 300, 400 pueden repetirse para producir claves públicas adicionales de primer usuario o claves públicas adicionales de segundo usuario.

Encriptación del ejecutable del software informático

El secreto común (SC) puede usarse como la base para generar una clave simétrica para la encriptación. En un ejemplo, el secreto común (SC) puede ser en la forma de un punto de curva elíptica (x_s, y_s). Este puede convertirse en un formato de clave estándar mediante el uso de funciones estándares acordadas por los nodos 3, 7. Por ejemplo, el valor x_s puede ser un entero de 256 bits que puede usarse como una clave para la encriptación AES_{256} (Estándar de Encriptación Avanzado). También puede convertirse en un entero de 160 bits mediante el uso de RIPEMD160. Los métodos 700, 800 de comunicación segura con encriptación por el primer usuario 23 se describirán ahora con referencia a la Figura 9.

En la realización provista más abajo a modo de ejemplo, el primer usuario 23 asociado al primer nodo 3 lleva a cabo el método 700 de encriptación del ejecutable del software informático. Se comprenderá que el método 700 puede igualmente aplicarse al segundo usuario 24 en el segundo nodo 7.

El primer usuario 23 determina 710 una clave simétrica según el secreto común (SC) determinado en el método 300, 400 de más arriba. Ello puede incluir convertir el secreto común (SC) en un formato de clave estándar según se describe más arriba. De manera similar, el segundo usuario 24 puede también determinar 810 la clave simétrica según el secreto común (SC).

La clave simétrica puede usarse por el primer usuario 23 para encriptar 720 el ejecutable del software informático para formar un ejecutable encriptado del software informático. El ejecutable encriptado del software informático se incluye 730 entonces en el cuerpo del software informático.

El software informático que comprende el ejecutable encriptado del software informático puede enviarse 740, en la red de comunicaciones 5, a una ubicación de almacenamiento. En un ejemplo, la ubicación de almacenamiento puede ser un depósito como, por ejemplo, una tabla de *hash* o la DHT 13. En otra ubicación, la ubicación de almacenamiento puede ser en Internet. En incluso otro ejemplo, la ubicación de almacenamiento puede ser un depósito basado en ordenador como, por ejemplo, un servidor, base de datos o instalación de almacenamiento provistos en un recurso basado en ordenador como, por ejemplo, el almacén de datos 17 asociado al primer dispositivo de procesamiento 21 del primer nodo 3.

El segundo usuario 24, a su vez, determina el ejecutable encriptado del software informático. La determinación del ejecutable encriptado del software informático puede comprender descargar el software informático de la ubicación de almacenamiento según se describe más arriba. En un ejemplo, el segundo usuario 24 descarga el ejecutable encriptado del software informático de una entrada en la DHT 13.

El segundo usuario 24 puede desencriptar 830 el ejecutable encriptado del software informático, con la clave simétrica, en el ejecutable del software informático.

Instalación del software informático

El ejecutable del software informático puede comprender instrucciones que hacen que un segundo dispositivo de procesamiento 27 asociado al segundo usuario 24 instale el software informático. En un ejemplo, el software informático se instala en el segundo dispositivo de procesamiento 27 después de que el segundo usuario 24 desencripta 830 el ejecutable encriptado del software informático.

En un ejemplo adicional, una clave de activación (AK) se determina a partir del segundo usuario 24 después de que el segundo usuario 24 desencripta 830 el ejecutable encriptado del software informático. En un ejemplo, una interfaz de usuario asociada al segundo dispositivo de procesamiento 27 puede motivar al segundo usuario 24 para la clave de activación (AK). El segundo usuario 24 puede proveer la clave de activación (AK) por un dispositivo de entrada asociado al segundo dispositivo de procesamiento 27 como, por ejemplo, un dispositivo de teclado, pantalla táctil o dispositivo de panel táctil, dispositivo de ratón o dispositivo de micrófono.

La clave de activación (AK) puede derivarse, de manera determinista, de una clave de activación de semilla. En un ejemplo, el primer usuario 23 puede determinar una clave de activación de semilla en el primer nodo 3. El primer usuario 23 puede entonces determinar un valor de generador (VG) según un mensaje. La clave de activación (AK) puede entonces determinarse según la clave de activación de semilla y el valor de generador (VG).

Determinación de un metadato (M) 910

Según se describe más arriba, el método 900 incluye determinar 910 un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido P2P 14. La determinación del metadato (M) puede comprender consultar el libro mayor distribuido P2P 14 para el registro de transacciones y recuperar el metadato (M) del registro de transacciones. La determinación del metadato (M) puede además comprender recibir el metadato (M) asociado a un registro de transacciones de un usuario, nodo o almacén de datos.

En un ejemplo, el segundo nodo 7 determina el metadato (M) mediante la consulta del libro mayor distribuido P2P 14 para acceder al registro de transacciones que se asocia al metadato (M). En un ejemplo adicional, puede accederse al registro de transacciones mediante el uso de una instalación públicamente disponible como, por ejemplo, www.blockchain.info.

En otro ejemplo, el primer nodo 3 u otro nodo no ilustrado de otra manera pueden consultar el libro mayor distribuido P2P 14 para el registro de transacciones. De esta manera, el metadato (M) puede recibirse en el segundo nodo 7 desde el primer nodo 3 u otro nodo no ilustrado de otra manera.

Según se describe más arriba, el metadato (M) puede incluirse en el registro de transacciones en, por ejemplo, uno o más de los lugares disponibles para las claves públicas en un *script* de rescate multifirma P2SH.

En otro ejemplo, el segundo nodo 7 puede recibir el metadato (M) de un depósito basado en ordenador como, por ejemplo, un servidor, base de datos o instalación de almacenamiento provistos en un recurso basado en ordenador. En un ejemplo adicional, se accede al metadato (M) desde el almacén de datos 17 asociado al primer dispositivo de procesamiento 21 del primer nodo 3.

Determinación de una indicación de una entrada almacenada en la DHT 920

El método 900 además incluye determinar 920 una indicación de una entrada almacenada en la DHT 13 del metadato (M). La determinación de una indicación de una entrada almacenada en la DHT 13 puede comprender recuperar la indicación de un campo en el metadato (M) del registro de transacciones asociado.

Según se describe más arriba, la Tabla 1 provee ejemplos de la información que puede estar comprendida en el metadato (M). La línea 2 de la Tabla 1 muestra un ejemplo de una indicación de una entrada almacenada en la DHT 13, donde la indicación es una dirección IP que identifica la ubicación de la entrada almacenada en la DHT 13. En un ejemplo, el segundo nodo 7 puede determinar la indicación de una entrada mediante la recuperación del valor asociado al campo LicencePointer del metadato (M).

En un ejemplo adicional, la indicación de una entrada puede almacenarse en un campo en el metadato (M) no ilustrado de otra manera.

Determinación de un tercer valor *hash* (H3) según el software informático 930

Según se describe más arriba, el método 900 además incluye determinar 930 un tercer valor *hash* (H3) según el software informático. La determinación de un tercer valor *hash* (H3) puede comprender calcular el tercer valor *hash* en el segundo nodo 7.

En un ejemplo, el segundo nodo 7 puede calcular el tercer valor *hash* (H3) según los contenidos del software informático. En un ejemplo adicional, el tercer valor *hash* (H3) puede basarse en el cuerpo del software informático. En incluso un ejemplo adicional, el tercer valor *hash* (H3) puede basarse en el ejecutable del software informático.

El tercer valor *hash* (H3) puede calcularse mediante el uso del algoritmo SHA-256 para crear una representación de 256 bits del software informático. Se apreciará que otros algoritmos pueden usarse, incluidos otros algoritmos en la familia SHA. Otros algoritmos de *hash* pueden incluir aquellos en la familia RIPEMD.

Determinación de un cuarto valor *hash* (H4) de la entrada en la tabla de *hash* distribuidas 940

El método 900 además incluye determinar 940 un cuarto valor *hash* (H4) de la entrada en la DHT 13. La determinación de un cuarto valor *hash* (H4) puede comprender acceder a la entrada en la DHT 13 donde se almacena el cuarto valor *hash* (H4).

El cuarto valor *hash* (H4) puede basarse en el software informático. En un ejemplo, el cuarto valor *hash* (H4) se basa en los contenidos de no encabezamiento del software informático. En otro ejemplo, el cuarto valor *hash* (H4) se basa en el ejecutable del software informático.

En un ejemplo, el cuarto valor *hash* (H4) se almacena en la entrada donde el metadato (M) asociado al registro de transacciones se almacena según se describe más arriba con referencia a 930. De esta manera, la determinación del cuarto valor *hash* (H4) puede comprender recuperar el cuarto valor *hash* (H4) de la entrada en la DHT 13 según se hace referencia por la indicación de una entrada. Es decir, por el campo LicencePointer del metadato (M).

En otro ejemplo, el cuarto valor *hash* (H4) es el primer valor *hash* (H1). Según se describe más arriba en el método 100, el primer valor *hash* (H1) se almacena en la DHT 13. De esta manera, el cuarto valor *hash* (H4) puede recuperarse por el segundo nodo 7 mediante la provisión a cualquier nodo participante de la DHT 13 de un mensaje de solicitud como, por ejemplo, *get(key),* donde *key* es la clave del par de valores de claves. En el presente ejemplo, *key* es el segundo valor *hash* (H2). El mensaje de solicitud puede enviarse a los nodos participantes hasta que se recibe por el nodo participante que se asigna al índice según se indica por la partición del espacio de claves.

Comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4) 950

Según se describe más arriba, el método 900 además incluye comparar 950 el tercer valor *hash* (H3) y el cuarto valor *hash* (H4). La comparación puede comprender determinar si el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) concuerdan.

En un ejemplo, una concordancia puede indicar que el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) son equivalentes.

Verificación de la integridad del software informático según la comparación 960

El método 900 además incluye verificar 960 la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4).

En un ejemplo, la verificación de la integridad del software informático ocurre si la comparación determina que el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) concuerdan.

Dispositivo de procesamiento

Según se describe más arriba, el primer nodo 3 y segundo nodo 7 pueden ser un dispositivo electrónico como, por ejemplo, un ordenador, tableta, dispositivo de comunicaciones móviles, servidor de ordenador, etc. El dispositivo electrónico puede incluir un dispositivo de procesamiento 21, 27, un almacén de datos 17 y una interfaz de usuario 15.

La Figura 10 ilustra un ejemplo de un dispositivo de procesamiento 21, 27. El dispositivo de procesamiento 21, 27 puede usarse en el primer nodo 3, segundo nodo 7 u otros nodos no ilustrados de otra manera. El dispositivo de procesamiento 21, 27 incluye un procesador 1510, una memoria 1520 y un dispositivo de interfaz 1540 que se comunican entre sí mediante un bus 1530. La memoria 1520 almacena un programa de software informático que comprende instrucciones legibles por máquina y datos para implementar el método 100, 300, 400, 600, 700 y 800 descrito más arriba, y el procesador 1510 lleva a cabo las instrucciones de la memoria 1520 para implementar el método 100, 300, 400, 600, 700 y 800. El dispositivo de interfaz 1540 puede incluir un módulo de comunicaciones que facilita la comunicación con la red de comunicaciones 5 y, en algunos ejemplos, con la interfaz de usuario 15 y periféricos como, por ejemplo, el almacén de datos 17. Debe notarse que aunque el dispositivo de procesamiento 1510 puede ser un elemento de red independiente, el dispositivo de procesamiento 1510 puede también ser parte de otro elemento de red. Además, algunas funciones llevadas a cabo por el dispositivo de procesamiento 1510 pueden distribuirse entre múltiples elementos de red. Por ejemplo, el primer nodo 3 puede tener múltiples dispositivos de procesamiento 21 para llevar a cabo el método 100, 300, 400, 600, 700 y 800 en una red de área local segura asociada al primer nodo 3.

Donde la presente descripción describe que un usuario, empleador, empleado, emisor, comerciante, proveedor u otra entidad lleva a cabo una acción particular (incluida la firma, emisión, determinación, cálculo, envío, recepción,

creación, etc.), dicha redacción se usa en aras de la claridad de presentación. Debe comprenderse que dichas acciones se llevan a cabo por los dispositivos informáticos operados por dichas entidades.

Las personas con experiencia en la técnica apreciarán que se pueden realizar numerosas modificaciones y/o variaciones en las realizaciones descritas más arriba, sin apartarse del alcance amplio general de la presente descripción. Las presentes realizaciones se considerarán, por lo tanto, en todos los aspectos, ilustrativas y no restrictivas.

REIVINDICACIONES

1. Un método implementado por ordenador para verificar la integridad de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares, el método comprendiendo:

determinar un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido entre pares;

determinar una indicación de una entrada almacenada en la tabla de *hash* distribuidas del metadato (M);

determinar un tercer valor *hash* (H3) según el software informático;

determinar un cuarto valor *hash* (H4) de la entrada en la tabla de *hash* distribuidas;

comparar el tercer valor *hash* (H3) y el cuarto valor *hash* (H4); y

verificar la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4).

2. El método de la reivindicación 1, en donde la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4) comprende determinar si el tercer valor *hash* (H3) y el cuarto valor *hash* (H4) concuerdan.

3. El método de la reivindicación 1 o 2, en donde antes de determinar el metadato (M), el método comprende:

determinar una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) de un registro de transacciones almacenado en el libro mayor distribuido entre pares;

determinar una segunda clave pública (P2) asociada al segundo usuario (U2) de una entrada almacenada en la tabla de *hash* distribuidas;

comparar la clave pública de segundo usuario (PU2) y la segunda clave pública (P2); y

verificar la propiedad del software informático según la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2).

4. El método de la reivindicación 3, en donde la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) comprende determinar si la clave pública de segundo usuario (PU2) y la segunda clave pública (P2) concuerdan.

5. El método de la reivindicación 3 o 4, en donde antes de determinar la clave pública de segundo usuario (PU2), el método comprende:

determinar un dato (D1) asociado al software informático;

determinar un primer valor *hash* (H1) según el software informático;

determinar un segundo valor *hash* (H2) según el dato (D1) y el software informático;

enviar, en una red de comunicaciones, el dato (D1), el primer valor *hash* (H1) y el segundo valor *hash* (H2) a una entrada para el almacenamiento en la tabla de *hash* distribuidas, en donde el segundo valor *hash* (H2) es una clave de un par de valores de claves y el dato (D1) y el primer valor *hash* (H1) son un valor en el par de valores de claves; y

determinar el metadato (M) que comprende el segundo valor *hash* (H2) para el almacenamiento en el libro mayor distribuido entre pares.

6. El método de cualquiera de las reivindicaciones precedentes, en donde el software informático comprende un encabezamiento y un cuerpo.

7. El método de la reivindicación 6, en donde el tercer valor *hash* (H3) se determina a partir del cuerpo del software informático.

8. El método de las reivindicaciones 6 o 7, en donde el encabezamiento comprende un valor *hash* del cuerpo del software informático.

9. El método de las reivindicaciones 6, 7 u 8, en donde el encabezamiento además comprende el segundo valor *hash* (H2).

10. El método de cualquiera de las reivindicaciones 6 a 9, en donde el cuerpo del software informático comprende un ejecutable del software informático.

11. El método de la reivindicación 10, en donde antes de determinar la clave pública de segundo usuario (PU2), el método comprende encriptar el ejecutable del software informático.

12. El método de la reivindicación 11, en donde encriptar el ejecutable del software informático comprende:

determinar un valor de generador (VG);

determinar una segunda clave pública de segundo usuario (P2U2) según la clave pública de segundo usuario (PU2) y el valor de generador (VG), en donde la segunda clave pública de segundo usuario (P2U2) forma un par criptográfico con una segunda clave privada de segundo usuario (V2U2);

determinar una segunda clave pública de primer usuario (P2U1) según la clave pública de primer usuario (PU1) y el valor de generador (VG), en donde la segunda clave pública de primer usuario (P2U1) forma un par criptográfico con una segunda clave privada de primer usuario (V2U1);

determinar un secreto común (SC) según la segunda clave pública de segundo usuario (P2U2) y la segunda clave privada de primer usuario (V2U1); y

encriptar el ejecutable del software informático con el secreto común (SC) para generar un ejecutable encriptado del software informático.

13. El método de la reivindicación 12, en donde el ejecutable encriptado del software informático se desencripta mediante:

la determinación del secreto común (SC) según la segunda clave pública de primer usuario (P2U1) y la segunda clave privada de segundo usuario (V2U2); y

la desencriptación del ejecutable del software informático con el secreto común (SC) para generar un ejecutable desencriptado del software informático.

14. El método de la reivindicación 13, que además comprende:

instalar el ejecutable desencriptado del software informático en un dispositivo de procesamiento asociado al segundo usuario (U2).

15. El método de la reivindicación 14, que además comprende:

determinar una clave de activación (AK) a partir del segundo usuario (U2); y

ejecutar instrucciones del ejecutable desencriptado del software informático según la clave de activación (AK).

16. Un programa de software informático que comprende instrucciones legibles por máquina para hacer que un dispositivo de procesamiento implemente el método de cualquiera de las reivindicaciones precedentes.

17. Un sistema informático para verificar la integridad de un software informático para la instalación mediante el uso de una tabla de *hash* distribuidas y un libro mayor distribuido entre pares, el sistema comprendiendo un dispositivo de procesamiento asociado a un nodo en una red de nodos entre pares, configurado para:

determinar un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido entre pares;

determinar una indicación de la ubicación de una entrada en la tabla de *hash* distribuidas del metadato (M);

determinar un tercer valor *hash* (H3) según el software informático;

determinar un cuarto valor *hash* (H4) de la entrada en la tabla de *hash* distribuidas;

comparar el tercer valor *hash* (H3) y el cuarto valor *hash* (H4); y

verificar la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4).

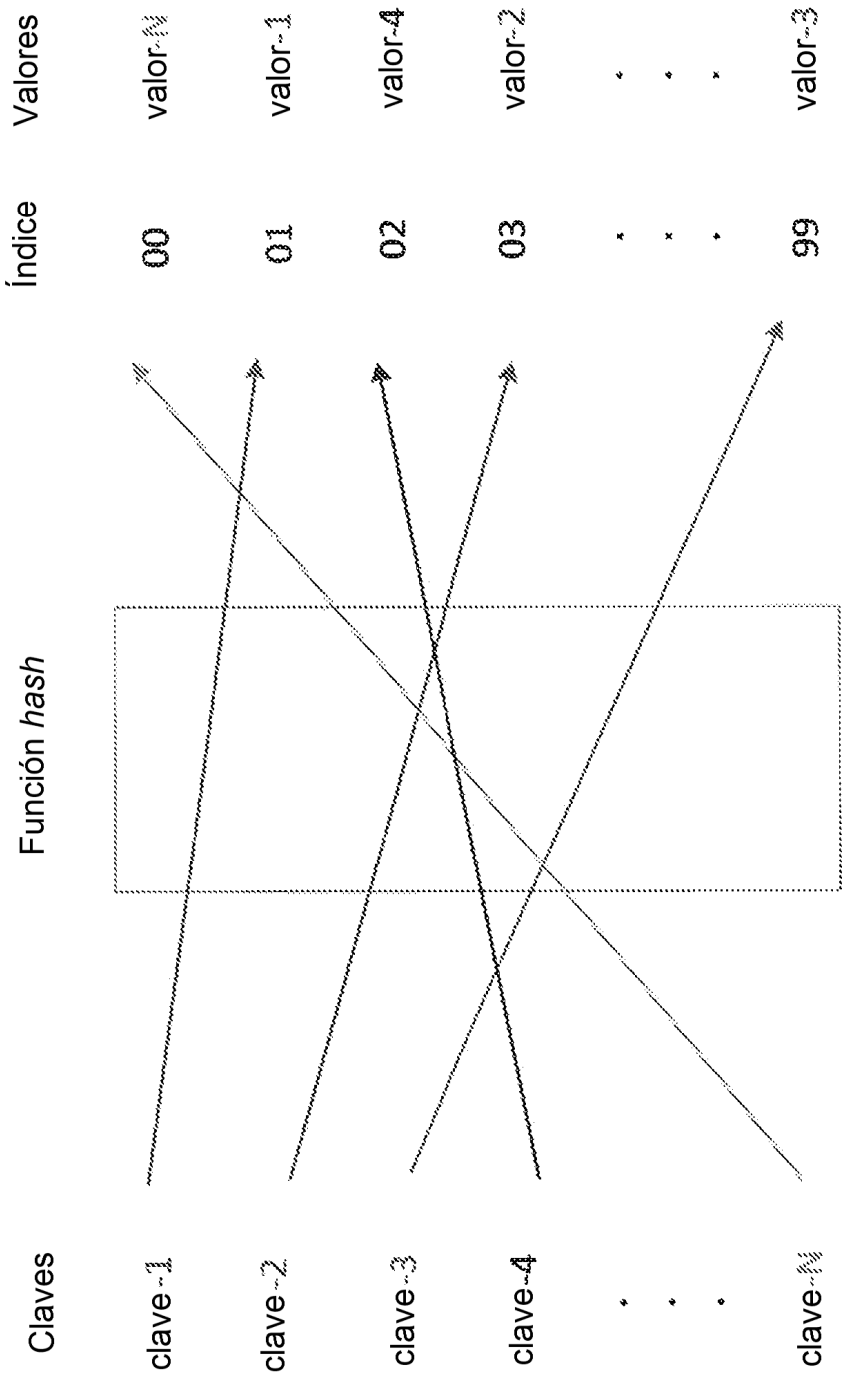

Claves
Función *hash*
Índice
Valores
clave-1
clave-2
clave-3
clave-4
clave-N
00
01
02
03
99
valor-N
valor-1
valor-4
valor-2
valor-3

Fig. 1

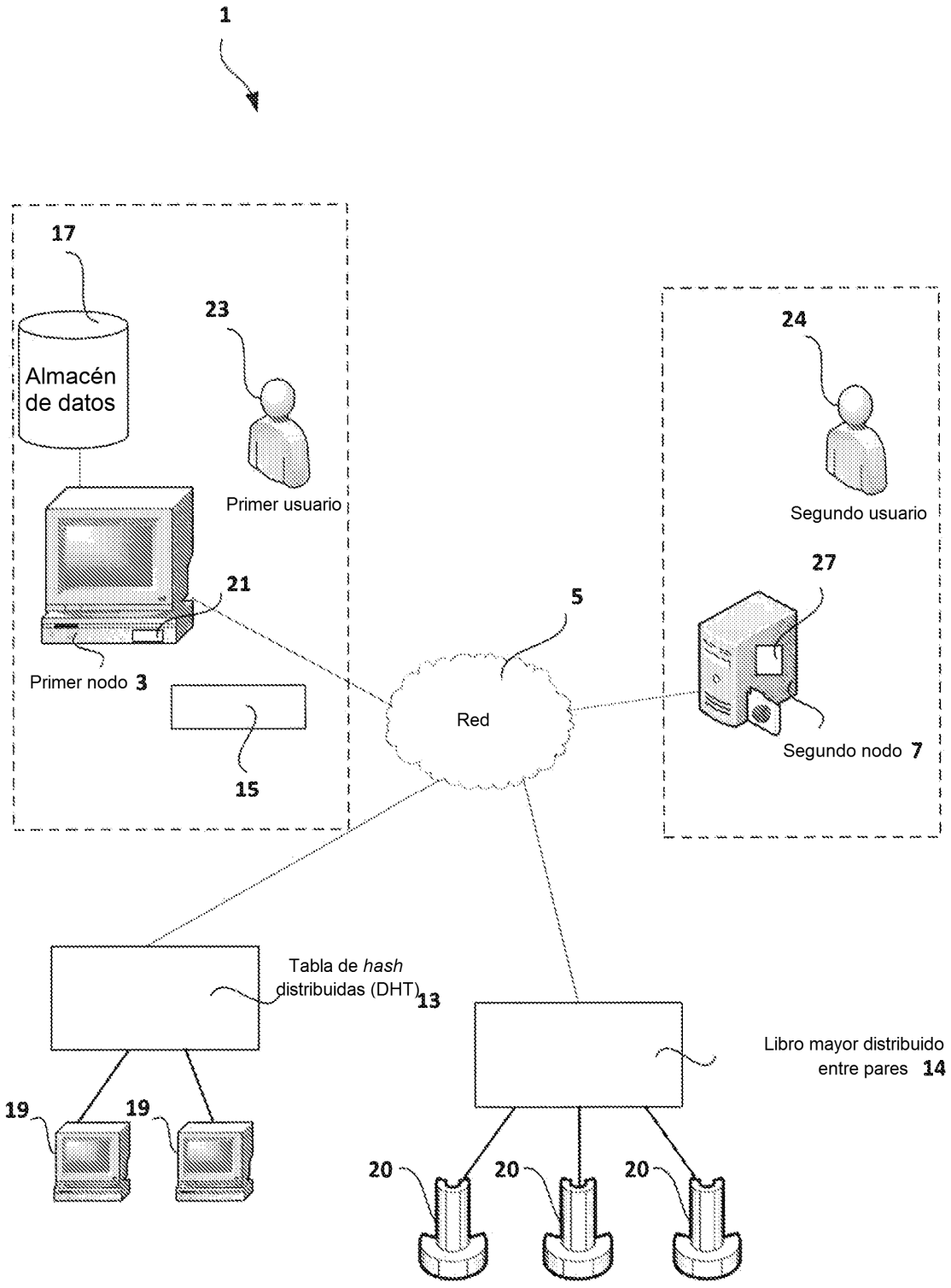

1
17
Almacén de datos
23
Primer usuario
24
Segundo usuario
27
21
5
Primer nodo 3
Red
Segundo nodo 7
15
Tabla de hash distribuidas (DHT) 13
Libro mayor distribuido entre pares 14
19
19
20
20
20

Fig. 2

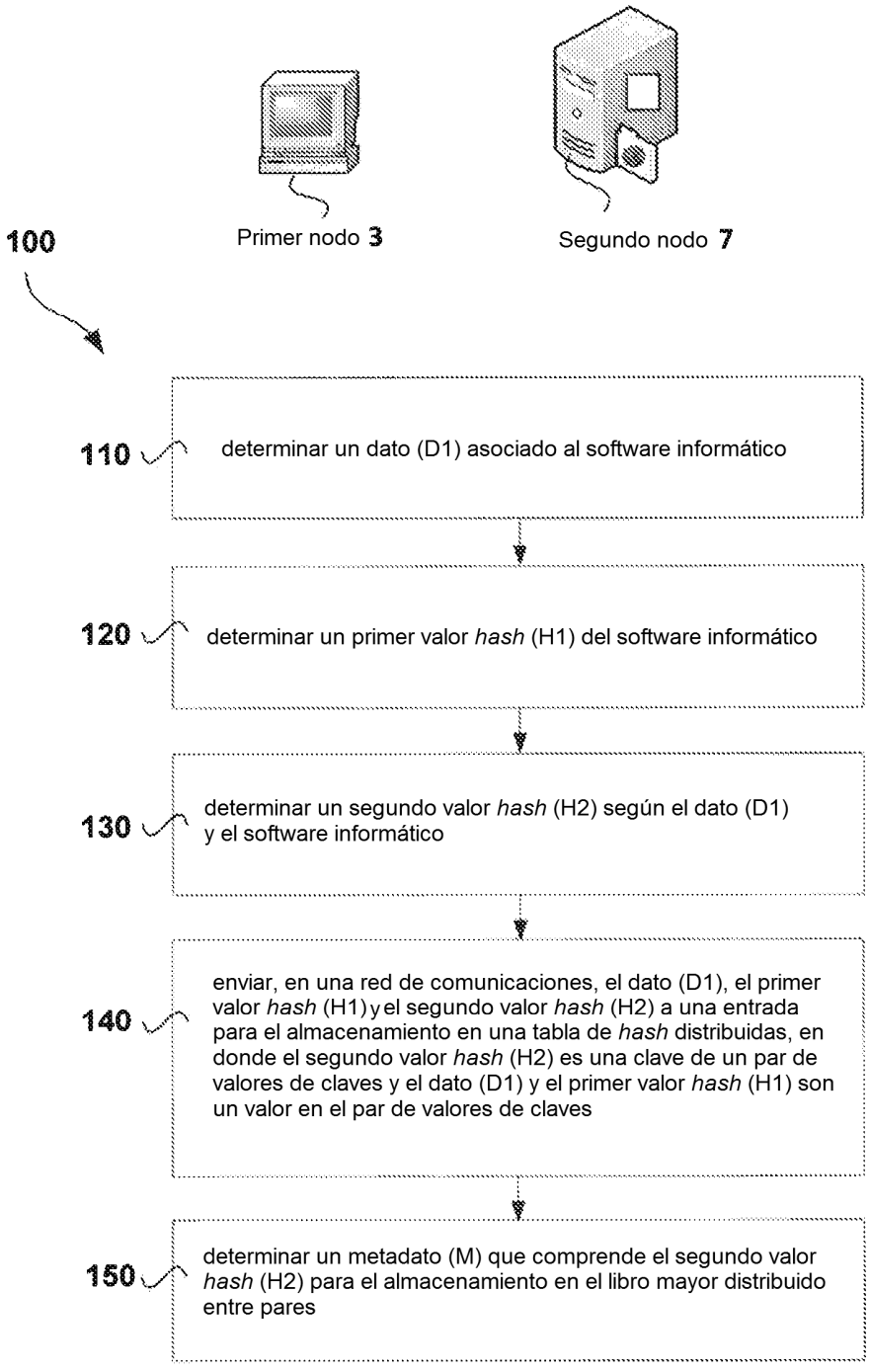

Primer nodo 3
Segundo nodo 7
100
110
determinar un dato (D1) asociado al software informático
120
determinar un primer valor *hash* (H1) del software informático
130
determinar un segundo valor *hash* (H2) según el dato (D1) y el software informático
140
enviar, en una red de comunicaciones, el dato (D1), el primer valor *hash* (H1) y el segundo valor *hash* (H2) a una entrada para el almacenamiento en una tabla de *hash* distribuidas, en donde el segundo valor *hash* (H2) es una clave de un par de valores de claves y el dato (D1) y el primer valor *hash* (H1) son un valor en el par de valores de claves
150
determinar un metadato (M) que comprende el segundo valor *hash* (H2) para el almacenamiento en el libro mayor distribuido entre pares

Fig. 3

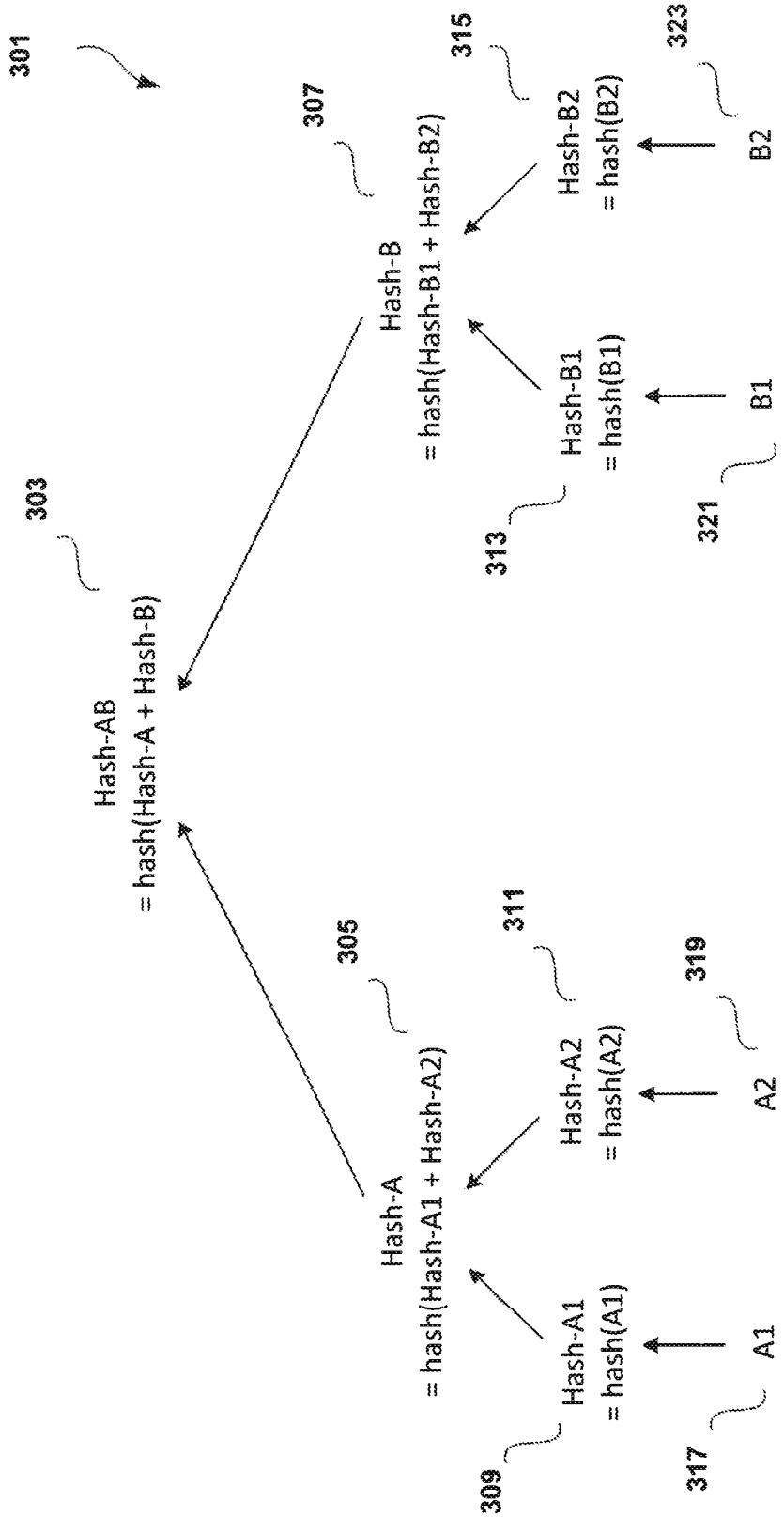

301
303
Hash-AB
= hash(Hash-A + Hash-B)
307
Hash-B
= hash(Hash-B1 + Hash-B2)
315
Hash-B2
= hash(B2)
323
B2
Hash-B1
= hash(B1)
313
321
B1
305
Hash-A
= hash(Hash-A1 + Hash-A2)
311
Hash-A2
= hash(A2)
319
A2
Hash-A1
= hash(A1)
309
317
A1

Fig. 4

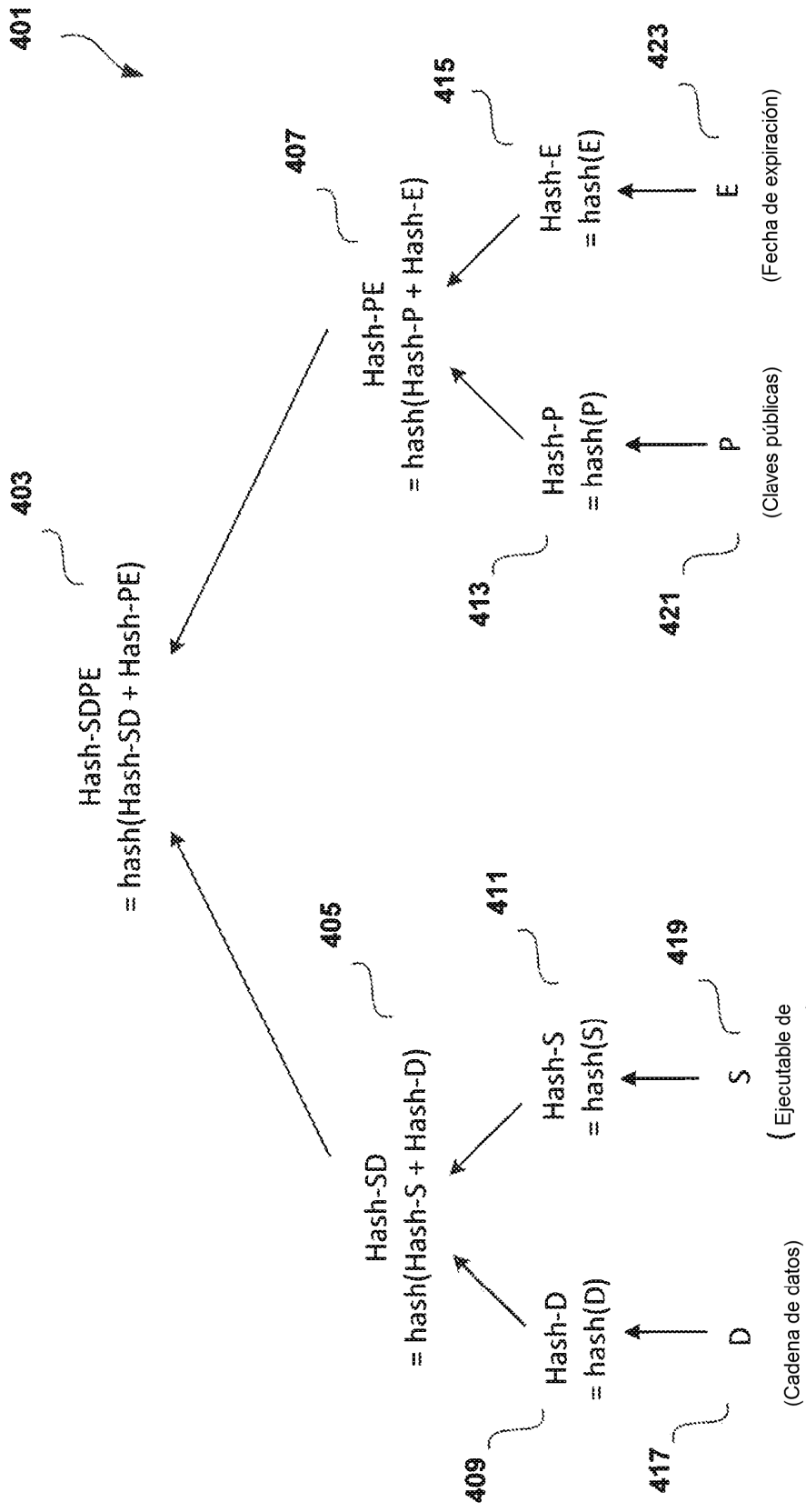
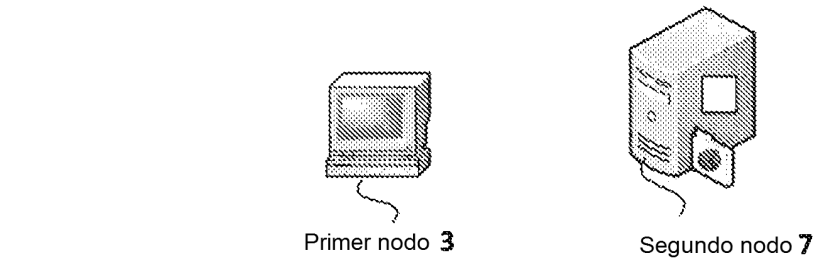

401
403
Hash-SDPE
= hash(Hash-SD + Hash-PE)
405
Hash-SD
= hash(Hash-S + Hash-D)
407
Hash-PE
= hash(Hash-P + Hash-E)
409
Hash-D
= hash(D)
411
Hash-S
= hash(S)
413
Hash-P
= hash(P)
415
Hash-E
= hash(E)
417
D
(Cadena de datos)
419
S
(Ejecutable de Software)
421
P
(Claves públicas)
423
E
(Fecha de expiración)

Fig. 5


Primer nodo 3
Segundo nodo 7

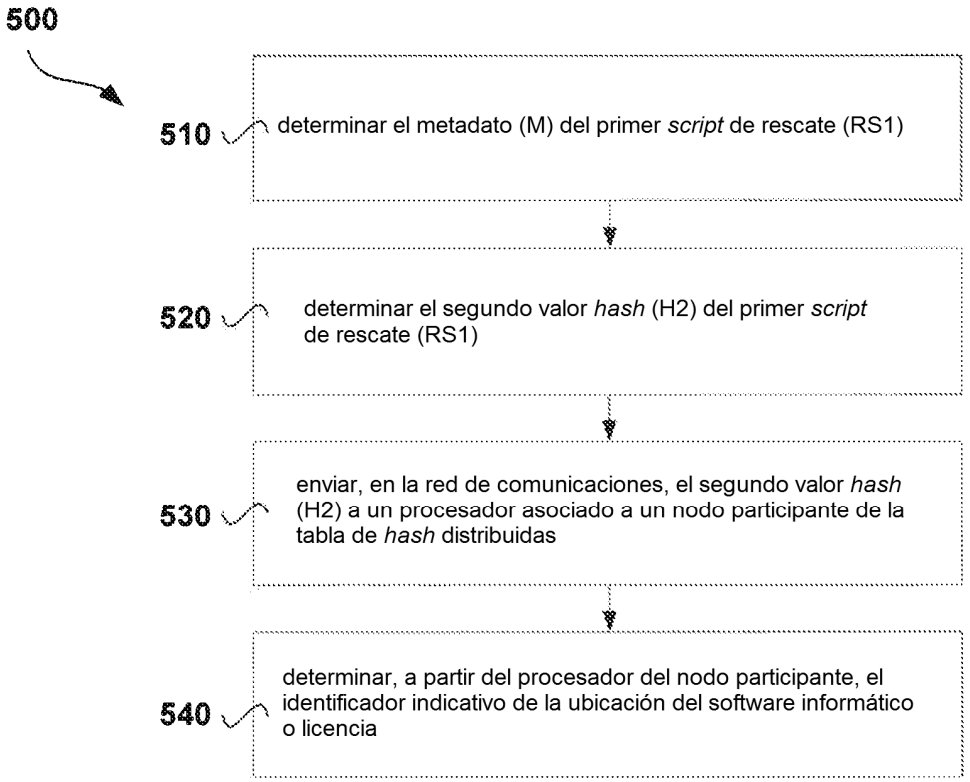

500
510
determinar el metadato (M) del primer *script* de rescate (RS1)
520
determinar el segundo valor *hash* (H2) del primer *script* de rescate (RS1)
530
enviar, en la red de comunicaciones, el segundo valor *hash* (H2) a un procesador asociado a un nodo participante de la tabla de *hash* distribuidas
540
determinar, a partir del procesador del nodo participante, el identificador indicativo de la ubicación del software informático o licencia

Fig. 6

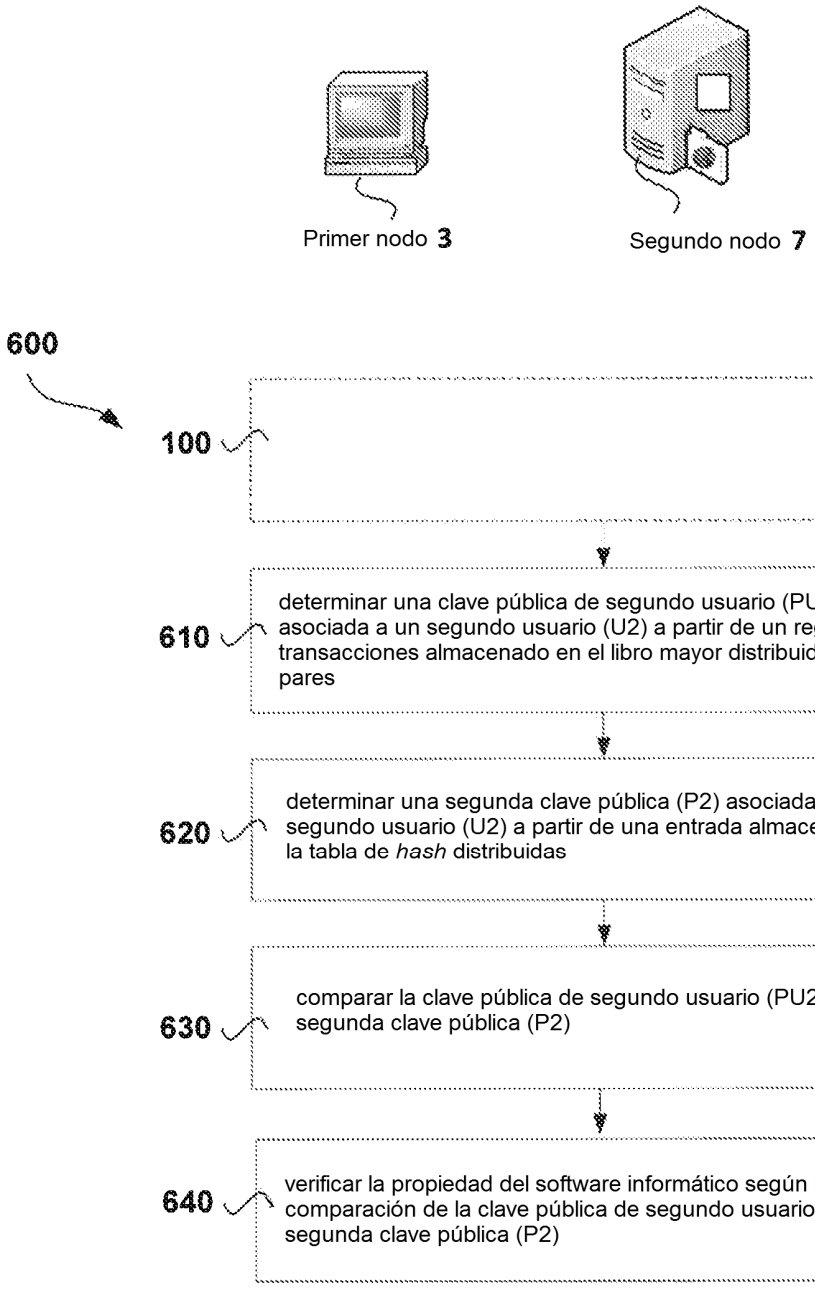

Primer nodo 3
Segundo nodo 7
600
100
610
determinar una clave pública de segundo usuario (PU2) asociada a un segundo usuario (U2) a partir de un registro de transacciones almacenado en el libro mayor distribuido entre pares
620
determinar una segunda clave pública (P2) asociada al segundo usuario (U2) a partir de una entrada almacenada en la tabla de *hash* distribuidas
630
comparar la clave pública de segundo usuario (PU2) y la segunda clave pública (P2)
640
verificar la propiedad del software informático según la comparación de la clave pública de segundo usuario (PU2) y la segunda clave pública (P2)

Fig. 7

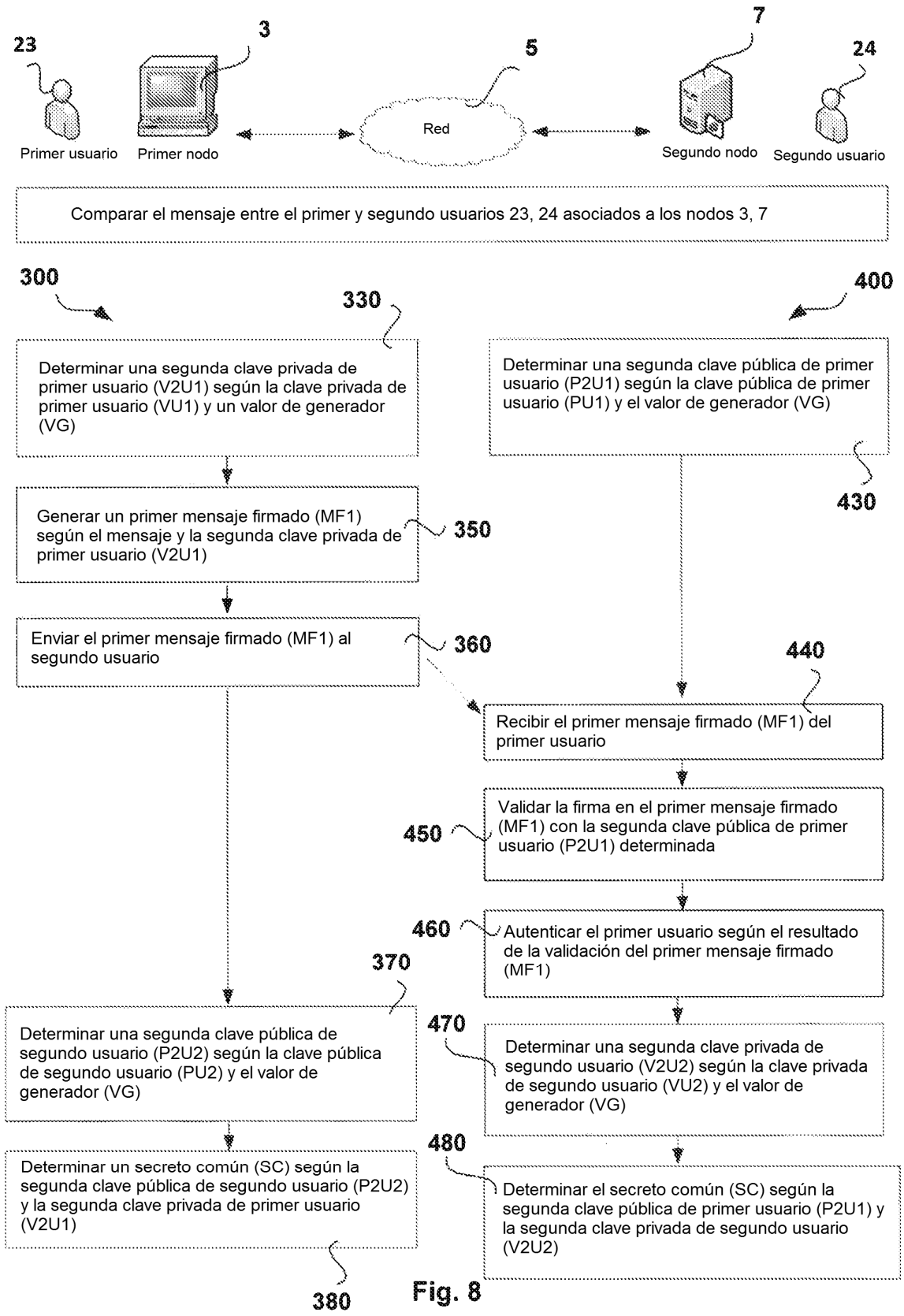

23
3
5
7
24
Primer usuario
Primer nodo
Red
Segundo nodo
Segundo usuario
Comparar el mensaje entre el primer y segundo usuarios 23, 24 asociados a los nodos 3, 7
300
330
400
Determinar una segunda clave privada de primer usuario (V2U1) según la clave privada de primer usuario (VU1) y un valor de generador (VG)
Determinar una segunda clave pública de primer usuario (P2U1) según la clave pública de primer usuario (PU1) y el valor de generador (VG)
430
Generar un primer mensaje firmado (MF1) según el mensaje y la segunda clave privada de primer usuario (V2U1)
350
Enviar el primer mensaje firmado (MF1) al segundo usuario
360
440
Recibir el primer mensaje firmado (MF1) del primer usuario
Validar la firma en el primer mensaje firmado (MF1) con la segunda clave pública de primer usuario (P2U1) determinada
450
460
Autenticar el primer usuario según el resultado de la validación del primer mensaje firmado (MF1)
370
Determinar una segunda clave pública de segundo usuario (P2U2) según la clave pública de segundo usuario (PU2) y el valor de generador (VG)
470
Determinar una segunda clave privada de segundo usuario (V2U2) según la clave privada de segundo usuario (VU2) y el valor de generador (VG)
480
Determinar un secreto común (SC) según la segunda clave pública de segundo usuario (P2U2) y la segunda clave privada de primer usuario (V2U1)
Determinar el secreto común (SC) según la segunda clave pública de primer usuario (P2U1) y la segunda clave privada de segundo usuario (V2U2)
380

Fig. 8

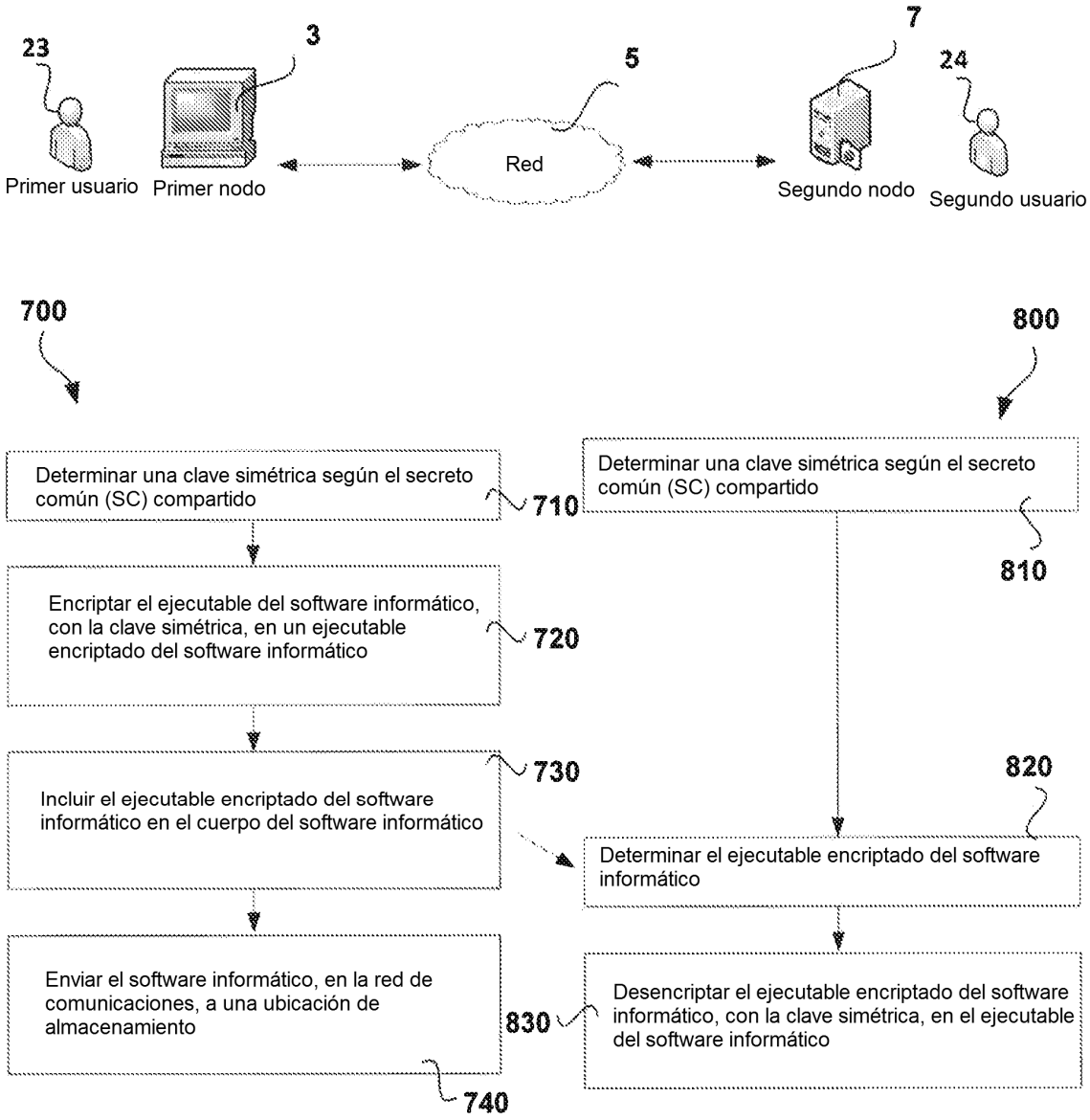

23
3
5
7
24
Primer usuario
Primer nodo
Red
Segundo nodo
Segundo usuario
700
800
Determinar una clave simétrica según el secreto común (SC) compartido
710
Determinar una clave simétrica según el secreto común (SC) compartido
810
Encriptar el ejecutable del software informático, con la clave simétrica, en un ejecutable encriptado del software informático
720
730
Incluir el ejecutable encriptado del software informático en el cuerpo del software informático
820
Determinar el ejecutable encriptado del software informático
Enviar el software informático, en la red de comunicaciones, a una ubicación de almacenamiento
740
830
Desencriptar el ejecutable encriptado del software informático, con la clave simétrica, en el ejecutable del software informático

Fig. 9

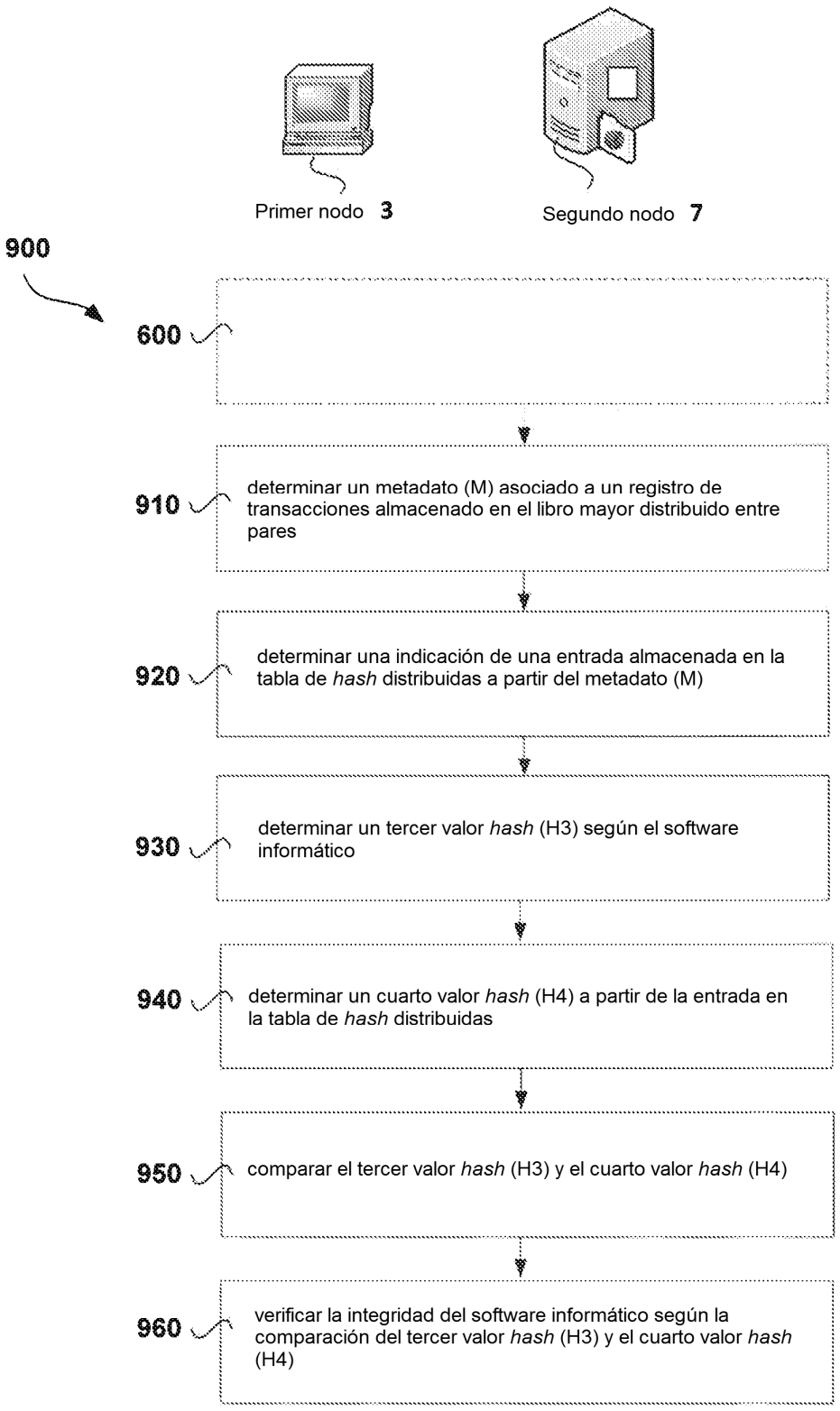

Primer nodo 3
Segundo nodo 7
900
600
910
determinar un metadato (M) asociado a un registro de transacciones almacenado en el libro mayor distribuido entre pares
920
determinar una indicación de una entrada almacenada en la tabla de *hash* distribuidas a partir del metadato (M)
930
determinar un tercer valor *hash* (H3) según el software informático
940
determinar un cuarto valor *hash* (H4) a partir de la entrada en la tabla de *hash* distribuidas
950
comparar el tercer valor *hash* (H3) y el cuarto valor *hash* (H4)
960
verificar la integridad del software informático según la comparación del tercer valor *hash* (H3) y el cuarto valor *hash* (H4)

Fig. 10

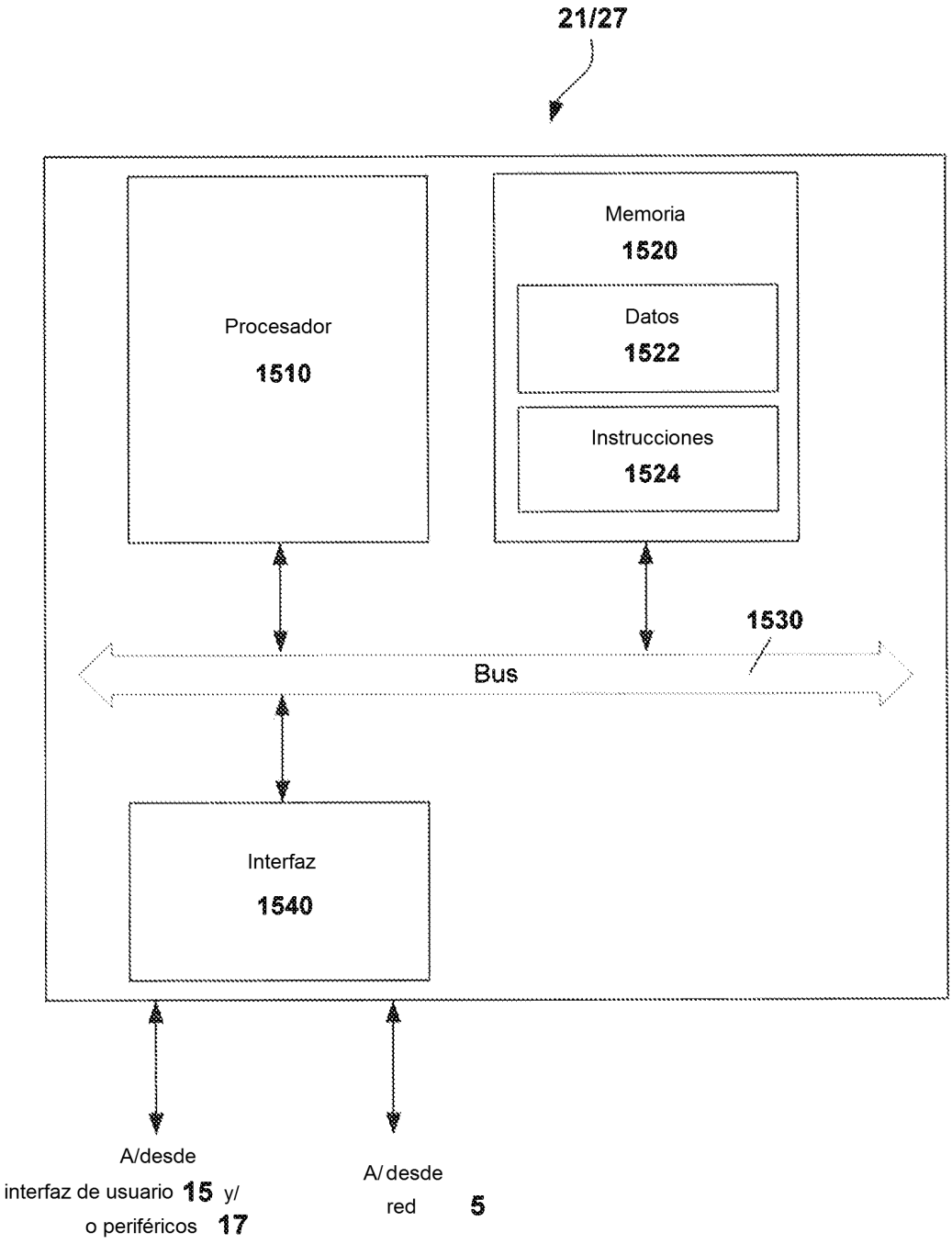

21/27
Procesador
1510
Memoria
1520
Datos
1522
Instrucciones
1524
1530
Bus
Interfaz
1540
A/desde
interfaz de usuario 15 y/
o periféricos 17
A/desde
red 5

Fig. 11