(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 612 180**

(51) Int. CI.:

G06F 11/07 (2006.01)
G06F 9/06 (2006.01)
G06F 11/14 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **11.01.2013 PCT/US2013/021145**

(87) Fecha y número de publicación internacional: **18.07.2013 WO2013106649**

(96) Fecha de presentación y número de la solicitud europea: **11.01.2013 E 13736355 (2)**

(97) Fecha y número de publicación de la concesión europea: **23.11.2016 EP 2802990**

(54) Título: **Tolerancia a fallos para operaciones complejas de computación distribuida**

(30) Prioridad:

13.01.2012 US 201261586472 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
12.05.2017

(73) Titular/es:

NETSUITE INC. (100.0%)
2955 Campus Drive Suite 100
San Mateo, CA 94403-2511, US

(72) Inventor/es:

PARRA. IVAN, OMAR y
WILLIAMS, DOUGLAS, H.

(74) Agente/Representante:

VALLEJO LÓPEZ, Juan Pedro

DESCRIPCIÓN

Tolerancia a fallos para operaciones complejas de computación distribuida

Antecedentes

Se ha generalizado el uso de ordenadores y redes de ordenadores para facilitar una amplia variedad de actividades como el trabajo y el esparcimiento. Las redes de ordenadores modernas incorporan capas de virtualización, de modo que los ordenadores físicamente remotos y los componentes informáticos puedan ser asignados a una tarea particular y luego reasignarse cuando se realiza la tarea. Los usuarios a veces hablan en términos de "nubes" de computación por la forma en que grupos de ordenadores y componentes informáticos pueden formarse y dividirse en respuesta a la demanda del usuario, y por que a menudo los usuarios nunca ven el hardware de computación que en última instancia ofrece los servicios de computación. Más recientemente, diferentes tipos de nubes de computación y servicios en la nube han comenzado a emerger.

Las plataformas de servicio en la nube varían en el tipo de servicios que prestan y en los tipos de aplicaciones que están destinadas a soportar. En un extremo del espectro se encuentran los servicios de "bajo nivel", tal como plataformas que proporcionan acceso al sistema operativo, uno o más marcos de desarrollo, bases de datos y otras instalaciones similares. Un objetivo primordial de estas plataformas es reducir el hardware y los costes de TI sin restringir de otra forma la elección del desarrollador de la aplicación de la solución técnica o espacio de aplicación. En el otro extremo del espectro están las plataformas que proporcionan instalaciones para crear aplicaciones en el contexto de una aplicación preexistente con un propósito bien definido. Este tipo de servicios en la nube de "alto nivel" suelen centrarse en una o más aplicaciones de usuario final bien definidas, tales como aplicaciones de negocios. Un objetivo de estas plataformas es permitir la creación de extensiones para una aplicación de núcleo. Los servicios prestados en este caso normalmente están orientados hacia el contexto de integrar la aplicación y lejos de servicios de bajo nivel y la elección de la solución técnica. Algunos servicios en la nube de alto nivel proporcionan una capacidad de personalizar y/o ampliar una o más de las aplicaciones de usuario final que proporcionan, sin embargo, servicios en la nube de alto nivel, no suelen proporcionar acceso directo a funciones de computación de bajo nivel. Esto puede ser problemático con respecto a la tolerancia a los fallos, por ejemplo, al mantenimiento de los datos y/o a la integridad de comportamiento después de experimentar un evento inesperado o de interrupción, tal como un fallo en la red de alimentación o de comunicaciones, ya que los enfoques convencionales suelen utilizar funciones de bajo nivel de computación para implementar la tolerancia a fallos.

La figura 1 representa aspectos de un entorno de computación 100 de ejemplo que puede beneficiarse de al menos una realización de la invención. Varias aplicaciones de cliente (no mostradas) que incorporan y/o están incorporadas en varios dispositivos de computación 104 pueden comunicarse con un servicio de computación distribuida de múltiples usuarios 108 a través de una o más redes 112. Ejemplos de dispositivos de computación adecuados 104 incluyen ordenadores personales, ordenadores servidores, ordenadores de sobremesa, ordenadores portátiles, asistentes digitales personales (PDA), teléfonos inteligentes, teléfonos celulares, ordenadores y productos electrónicos de consumo que incorporan uno o más componentes del dispositivo de computación, tal como uno o más procesadores. Ejemplos de redes adecuadas incluyen redes de comunicación, incluyendo tecnologías cableadas e inalámbricas, redes que operan de acuerdo con cualquier establecimiento de una red adecuada y/o protocolo de comunicación, intranets privadas y/o Internet.

El servicio de computación distribuida de múltiples inquilinos 108 puede incluir múltiples niveles de procesamiento que incluyen un nivel de interfaz de usuario 116, un nivel de aplicación 120 y una capa de almacenamiento de datos 124. El nivel de interfaz de usuario 116 puede mantener múltiples interfaces de usuario 128, incluidas interfaces gráficas de usuario y/o interfaces basadas en web. Las interfaces de usuario 128 pueden incluir una interfaz de usuario predeterminada para el servicio, así como una o más interfaces de usuario personalizadas mediante uno o más inquilinos del servicio. La interfaz de usuario por defecto puede incluir componentes que permiten a los inquilinos mantener interfaces de usuario personalizadas, así como administrar su participación en el servicio. Cada nivel puede implementarse mediante un conjunto distribuido de ordenadores y/o componentes informáticos, incluyendo servidores de ordenador. El nivel de almacenamiento de datos 124 puede incluir un almacén de datos de servicio central 132, así como un almacén de datos (o almacenes de datos) 136 para almacenar datos de inquilinos.

El nivel de aplicación 120 del servicio de computación distribuida de múltiples inquilinos 108 puede proporcionar servidores de aplicaciones 140 para ejecutar aplicaciones de usuario final personalizables y/o extensibles. Por ejemplo, el nivel de aplicación puede permitir la personalización con un lenguaje de programación, tal como un lenguaje de secuencia de comandos. El código de programa personalizado puede ejecutarse en un entorno de ejecución controlado de instancia 144 de los servidores de aplicaciones 140. Por ejemplo, las secuencias de comandos personalizadas pueden ser ejecutadas por un intérprete de lenguaje de secuencia de comandos.

Intentos convencionales para permitir la personalización de los servicios en la nube de alto nivel, tales como el servicio de computación distribuida de múltiples inquilinos se muestran en la figura 1, mientras que abordar cuestiones de tolerancia a fallos son ineficientes, ineficaces y/o tienen efectos secundarios indeseables u otros inconvenientes.

Por ejemplo, los programas que se ejecutan en un sistema operativo de ordenador convencional normalmente pueden utilizar mecanismos de bajo nivel proporcionados por una base de datos para asegurarse de que se mantiene la consistencia de datos en presencia de interrupciones inesperadas. Para los procesos de ejecución particularmente larga, se pueden emplear otros mecanismos para realizar el seguimiento del progreso de un programa para soportar la recuperación de eventos inesperados, tal como la pérdida de energía, interrupciones de red u otros fallos del sistema. Por ejemplo, un programa que realiza una operación repetida en una lista homogénea de objetos de datos puede encapsular cada operación idéntica en una transacción de la base de datos e incluir información para indicar la finalización de cada unidad de trabajo. Si se interrumpe el proceso, cuando se reinicia el sistema, el programa puede preguntar para los objetos sin procesar y reanudar sin sacrificar la coherencia de los datos.

Un proceso que se ejecuta en una plataforma basada en la nube no puede, sin embargo, tener acceso a las mismas instalaciones de bajo nivel a disposición de uno escritas directamente en el sistema operativo. En un sistema de este tipo, la consistencia de los datos solo puede garantizarse dentro de un ámbito de una operación de acceso de datos a nivel del sistema, tales como la lectura o la escritura de un objeto de negocio. Incluso en un caso en el que un proceso de larga duración está construido a partir de múltiples unidades de computación idénticas, estas unidades pueden incluir más de una única operación de acceso a datos. Esto puede hacer que el proceso vulnerable a inconsistencias en los datos si se produce una interrupción inesperada, no entre unidades de computación, pero en medio de una sola unidad computacional.

Los usuarios de plataformas de bajo nivel están en condiciones de gestionar problemas de coherencia, ya que normalmente tienen acceso a instalaciones a disposición de los entornos de desarrollo convencionales (no en la nube, en las instalaciones, etc.). Por ejemplo, la naturaleza transaccional de una base de datos relacional se puede combinar con una arquitectura que minimiza un número de operaciones realizadas en una sola transacción como la base para asegurar un estado de los datos uniforme duradero.

En plataformas de alto nivel, sin embargo, un programa puede no tener acceso directo a una transacción de base de datos. En lugar de ello, las transacciones de base de datos subyacentes pueden utilizarse para garantizar la coherencia de los datos de acceso a los objetos de negocio de alto nivel con los que interactúan estas plataformas. Para crear un proceso atómico que abarca múltiples operaciones de acceso a datos de alto nivel, estas plataformas pueden proporcionar una forma restringida de gestión de transacciones que se extiende en un pequeño número de accesos y/o imponer más límites a los tipos de servicios de plataforma que se pueden utilizar durante la transacción.

Esta plataforma de alto nivel, tal como el servicio de computación distribuida de múltiples inquilinos que se muestra en la figura 1, puede ser susceptible a interrupciones intermitentes, incluyendo fallo del sistema, un mecanismo de gestión de recursos, reinicio del sistema planeado, y reinicio de monitorización iniciada. La gobernabilidad del recurso puede incluir recursos de seguimiento de utilización en función de cada inquilino mediante la plataforma y la limitación de la utilización de los recursos de conformidad con un acuerdo entre el inquilino y el proveedor de servicios de plataforma. Sin embargo, ventajosamente en el proveedor de servicios, las interrupciones no necesitan garantizarse para proporcionar señales que permiten cerrar de modo seguro un proceso de cliente. Estas interrupciones se pueden clasificar como (i) las que se producen durante una operación de escritura de datos y (ii) los que se producen fuera de una operación de escritura de datos. En el primer caso, las interrupciones que se producen dentro de una operación de escritura de datos no pueden salir de un objeto de negocio en un estado inconsistente, ya que están protegidas por una transacción de base de datos subyacente. Sin embargo, durante el último tipo de interrupción, se puede perder la consistencia de datos entre los objetos de negocio.

El documento US2008/0307258, que se considera que representa la técnica anterior más próxima, describe un método para la recuperación de una instancia de un gestor de trabajos que se ejecuta en uno de una pluralidad de nodos utilizados para ejecutar los elementos de procesamiento asociados a los trabajos que se ejecutan dentro de un sistema de procesamiento de flujo de datos cooperativo. Los estados de los elementos de procesamiento se comprueban apuntando a un mecanismo de persistencia en comunicación con el gestor de trabajos, ejecutándose el gestor de trabajos en el sistema distribuido global en uno de los nodos del sistema distribuido.

Otros sistemas de la técnica anterior se divulgan en el documento US2002/0087657, que divulga una arquitectura de ordenador distribuida sin estado con objetos de estado de almacenamiento en caché orientados a servidores mantenidos en la red o el cliente, y el documento US2004/0193941, que divulga un método para el soporte asíncrono de comunicación tolerante a fallos y adaptativo.

Las realizaciones de la invención se dirigen hacia la solución de estos y otros problemas de forma individual y colectivamente.

Sumario

Este sumario es una vista general de alto nivel de diversos aspectos de los presentes métodos y sistemas para habilitar una tolerancia a fallos en un sistema de computación distribuida que ejecuta un proceso de cliente e introduce algunos de los conceptos que se describen con más detalle en la sección de la descripción detallada a

continuación. Este sumario no pretende identificar las características clave o esenciales de la materia reivindicada, ni está destinado a ser utilizado para determinar el alcance de la materia reivindicada.

De acuerdo con un primer aspecto de la presente invención, se proporciona un método para permitir una tolerancia a fallos de acuerdo con la reivindicación independiente 1 adjunta.

La presente invención proporciona además un sistema para habilitar una tolerancia a fallos de acuerdo con la reivindicación independiente 4.

En particular, varios aspectos de las realizaciones descritas a continuación se dirigen a métodos que puede incluir las etapas de instanciar un entorno de ejecución en relación a dicho proceso de cliente y ejecutar instrucciones dentro de dicho entorno de ejecución. Las instrucciones ejecutadas, a su vez, hacen que el entorno de ejecución emita instrucciones adicionales al sistema de computación distribuida en relación con las acciones a realizar con respecto a datos almacenados en el sistema de computación distribuida. Esas instrucciones son recibidas por un proxy de interfaz de objetos y al menos una de las instrucciones es una instrucción de guardar el estado, que hace que el proxy de interfaz de objetos guarde un estado actual del entorno de ejecución en un almacén de datos.

Otros aspectos de las diversas realizaciones descritas a continuación se dirigen a sistemas que pueden incluir un entorno de ejecución instanciado en un sistema de computación distribuida; un proxy de interfaz de objetos que tiene un módulo de tolerancia a fallos y también se ejecuta en el sistema de computación distribuida; y un almacén de datos. El entorno de ejecución puede ejecutar instrucciones en al menos un objeto de software de acuerdo con un proceso de cliente y el proxy de interfaz de objetos actúa como una interfaz para el encaminamiento de esas instrucciones desde el entorno de ejecución al objeto de software. El módulo de tolerancia a fallos puede recibir una instrucción de guardar el estado del entorno de ejecución, lo que hará que el módulo de tolerancia a fallos guarde un estado actual de dicho entorno de ejecución al almacén de datos.

Otros aspectos de las realizaciones descritas a continuación se dirigen a métodos que puede incluir las etapas de instanciar un entorno de ejecución en relación a un proceso de cliente que funciona en un sistema de computación distribuida y ejecutar instrucciones dentro del entorno de ejecución. Las instrucciones hacen que el entorno de ejecución emita instrucciones adicionales en relación con las acciones a realizar con respecto a datos almacenados en el sistema de computación distribuida. Un proxy de interfaz de objetos recibe y monitoriza esas instrucciones y determina si el entorno de ejecución está en una condición que guarda el estado deseado. Si es así, el estado actual del entorno de ejecución se guarda en un almacén de datos.

Otros aspectos de las diversas realizaciones descritas a continuación se dirigen a sistemas que pueden incluir un entorno de ejecución instanciado en un sistema de computación distribuida; un proxy de interfaz de objetos que tiene un módulo de tolerancia a fallos y también se ejecuta en el sistema de computación distribuida; y un almacén de datos. El entorno de ejecución puede ejecutar instrucciones en al menos un objeto de software de acuerdo con un proceso de cliente y el proxy de interfaz de objetos actúa como una interfaz para el encaminamiento de esas instrucciones desde el entorno de ejecución al objeto de software. El módulo de tolerancia a fallos está configurado para monitorizar cada instrucción recibida por el módulo de tolerancia a fallos y determinar si la ejecución de la instrucción coloca el entorno de ejecución en una condición de guardar el estado deseado y, si es así, guardar el estado actual del entorno de ejecución en el almacén de datos.

Otros objetos y ventajas de la presente invención serán evidentes para los expertos ordinarios en la técnica a partir de una revisión de la descripción detallada de la presente invención y las figuras incluidas.

Breve descripción de los dibujos

Se describirán varias realizaciones de acuerdo con la presente divulgación con referencia a los dibujos, en los que:

la figura 1 es un diagrama esquemático que muestra aspectos de un entorno de computación de ejemplo de acuerdo con al menos una realización de la invención;

la figura 2 es un diagrama esquemático que representa aspectos de una plataforma de servicio de computación distribuida de alto nivel de ejemplo de acuerdo con al menos una realización de la invención;

la figura 3 es un diagrama de interacción que representa aspectos de interacciones de ejemplo de acuerdo con al menos una realización de la invención;

las figuras 4a y 4b son diagramas de flujo de proceso de alto nivel que representan aspectos de guardar el estado y restaurar las operaciones del estado de acuerdo con al menos una realización de la invención;

las figuras 5a y 5b son diagramas de flujo de proceso de alto nivel que representan aspectos de guardar el estado y restaurar las operaciones del estado de acuerdo con al menos una realización alternativa de la invención;

la figura 6 es un diagrama de flujo de proceso de alto nivel que representa aspectos de guardar el estado y restaurar las operaciones del estado de acuerdo con al menos una realización alternativa de la invención;

la figura 7 es un diagrama de flujo que representa etapas de ejemplo para tolerancia a fallos de acuerdo con al menos una realización de la invención;

la figura 8 es un diagrama esquemático que muestra aspectos de un ordenador de ejemplo de acuerdo con al menos una realización de la invención.

Debe tenerse en cuenta que los mismos números se utilizan en toda la descripción y en las figuras para hacer referencia a los mismos componentes y características.

De acuerdo con al menos una realización de la invención, está habilitada la tolerancia a fallos para las operaciones de computación distribuidas complejas. El mecanismo de guardar el estado puede persistir el entorno de ejecución controlada, y el estado guardado puede restaurarse si el entorno de ejecución controlado detecta que se ha producido una interrupción. El mecanismo de guardar el estado puede actuar en respuesta a una llamada explícita de "guardar el estado" o un mensaje y/o cuando se detecta un estado de ejecución adecuado. El control sobre el mecanismo de guardar el estado puede estar expuesto a los desarrolladores de código de programa personalizado, por ejemplo, con una interfaz de programación.

De acuerdo con al menos una realización de la invención, la plataforma proporciona acceso a la funcionalidad de la plataforma con objetos de negocio que tiene interfaces de programación (a veces denominadas interfaces de programación de aplicaciones o APIs). En un entorno de computación distribuida, se puede acceder a tales interfaces, por ejemplo, con llamadas de función adecuadas, llamadas a funciones remotas y/o protocolos de mensajería. La plataforma puede garantizar la integridad de las operaciones de elemento de interfaz individual, pero no operaciones necesariamente complejas y/o compuestas que implican la activación de múltiples elementos de interfaz. Un monitor de acceso a interfaz de objeto de negocio puede actuar como un enrutador de llamada o proxy para dicho acceso de interfaz. Además, el monitor puede incorporar el mecanismo de guardar el estado, y hacerlo accesible con una interfaz de programación, por ejemplo, teniendo los elementos de "guardar el estado" y "restaurar el estado".

De acuerdo con al menos una realización de la invención, un intérprete de secuencia de comandos puede ejecutar una secuencia de comandos que causa múltiples transacciones atómicas, simples y/o integrales (colectivamente, "atómico") con respecto a un conjunto de objetos de negocio. Un componente de monitor puede monitorizar acciones de secuencia de comandos con respecto al conjunto de objetos de negocio y puede guardar los estados del intérprete de secuencia de comandos y/o los objetos de negocio como un punto de recuperación para que la secuencia de comandos se pueda reanudar en el punto de recuperación en caso de que se interrumpa el conjunto de múltiples transacciones atómicas. Los puntos de recuperación pueden crear instrucciones explícitas adecuadas mediante la secuencia de comandos y/o basadas al menos en parte en la acción de la secuencia de comandos monitorizado con respecto a los objetos de negocio.

La figura 2 representa aspectos de una plataforma de servicio de computación distribuida de alto nivel 200 de ejemplo de acuerdo con al menos una realización de la invención. La plataforma puede proporcionar servicios y recursos compartidos para permitir procesos de cliente 204 para interactuar con una aplicación incorporada organizada de múltiples inquilinos 208. El proceso de cliente 204 puede interactuar con la aplicación integrada 208 a través de una interfaz de programación de aplicaciones (API) 212 que incluye métodos 215 para buscar, leer y escribir datos, métodos para comunicarse con sistemas externos y otros métodos de utilidad para cálculos en el proceso (acceso sin datos). El intercambio de información entre el almacén de datos de aplicación integrada 216 y el proceso de cliente 204 puede producirse a través de la capa de objeto de negocio 218. Por ejemplo, los objetos de negocio 220 pueden corresponder a múltiples tablas de bases de datos subyacentes 224. Cada operación de acceso de datos proporcionada por el API puede ventajosamente tener una integridad garantizada, por ejemplo, correspondiente a las propiedades de atomicidad, consistencia, aislamiento y durabilidad (ACID) de transacciones de base de datos. Sin embargo, la plataforma 200 puede ser susceptible a las interrupciones intermitentes, incluyendo fallo del sistema, un mecanismo de gestión de recursos, reinicio del sistema planificado, y reinicio de monitorización iniciado.

La plataforma puede proporcionar una aplicación de alto nivel 208, tal como una aplicación de negocio, al menos en parte con un conjunto de objetos de negocio 220 en la capa de objetos de negocio 218. La aplicación de alto nivel 208 puede personalizarse mediante inquilinos del servicio con recursos gestionados de inquilinos, que incluyen ajustes personalizados, código de programa a medida, tales como secuencias de comandos, módulos de programas a medida, y cualquiera de los componentes de configuración personalizados adecuados. Los entornos de ejecución 236 pueden instanciarse para el código de programa personalizado y/o módulos de programas personalizados. Por ejemplo, cuando el código de programa personalizado incluye código escrito utilizando un lenguaje de programación interpretado como un lenguaje de secuencia de comandos, un intérprete 240 puede instanciar entornos de ejecución 236 para secuencias de comandos y/o tareas o trabajos asociados.

Por ejemplo, el intérprete 240 puede instanciar un entorno de ejecución 236 para una secuencia de comandos. El intérprete 240 puede luego ejecutar la secuencia de comandos en el contexto del entorno de ejecución instanciado 236. La secuencia de comandos puede causar uno o más mensajes de aplicación (por ejemplo, llamadas y/o mensajes de interfaz de objetos de negocio) entre el entorno de ejecución 236 y la capa de objetos de negocio 220 para ser recibidos y/o interceptados mediante un proxy de interfaz de objetos 248. Los mensajes pueden ser encaminados a continuación, al objeto de negocio 220 apropiado mediante el proxy 248.

El código personalizado que funciona en la aplicación de entornos de ejecución 236 puede acceder a las interfaces de objetos de negocio 252 a través de un proxy de interfaz de objetos 248. Por ejemplo, el proxy de interfaz de objetos 248 puede ser un "proxy fino" que simplemente monitoriza las llamadas funcionales remotas y/o mensajes de protocolo asociados (colectivamente "llamadas"), un balanceo de carga o un proxy de enrutamiento que distribuye la carga de llamadas, y/o un almacenamiento en caché. De acuerdo con realizaciones ilustrativas de los presentes métodos y sistemas, el proxy de interfaz de objetos 248 puede incorporar además un módulo de tolerancia a fallos 256. Por ejemplo, el módulo de tolerancia a fallos 256 puede proporcionar los elementos de interfaz "guardar el estado" y "restaurar el estado" 260, 264. La activación del elemento de interfaz "guardar el estado" 260 con el código del programa en un entorno de ejecución de aplicaciones 236 puede resultar en una "instantánea" restaurable del estado del entorno de ejecución que se guarda en un almacén de datos. La activación del elemento de la interfaz "restaurar el estado" 260 con respecto a un entorno de ejecución de aplicaciones 236 puede resultar en el entorno de ejecución de aplicaciones que se restaura a un estado que corresponde a una "instantánea" previamente guardada. Como se describe a continuación, el módulo de tolerancia a fallos 256 también puede monitorizar llamadas para detectar los momentos óptimos y/o prácticos para guardar el estado de las instancias de entorno de ejecución de aplicaciones.

De acuerdo con al menos una realización de los presentes métodos y sistemas, la funcionalidad de guardar el estado del proxy de la interfaz de objetos 248 puede capturar un estado interno de un proceso que se está ejecutando actualmente, tal como una secuencia de comandos que se ejecuta en un entorno de ejecución 236, encapsulando todas las llamadas nativas hechas por el proceso a una capa de API delgada. Por ejemplo, muchos motores de intérprete de lenguaje de secuencia de comandos incluyen la capacidad nativa de capturar el estado actual de ejecución de un proceso. De acuerdo con las realizaciones de los presentes métodos y sistemas, esta capacidad puede ser aprovechada para serializar los datos binarios que representan la pila de ejecución de los procesos, incluyendo cualesquiera datos residuales del intérprete y nativos de la aplicación, y escribirlos en un almacén de datos. Realizaciones de los presentes métodos y sistemas permiten así una preservación del estado de ejecución actual, así como un punto de recuperación fiable en el caso de que se produzca un fallo entre llamadas al API.

Las capas y/o componentes de la plataforma de servicios de computación distribuida pueden implementarse, al menos en parte, con los almacenes de datos y/o recursos de computación (por ejemplo, recursos del sistema operativo del ordenador) en una capa de almacenamiento de datos y una capa del sistema operativo del ordenador.

Las figuras 3 a 6 ilustran varios aspectos de interacciones de componentes de plataforma de ejemplo de acuerdo con al menos una realización de los presentes métodos y sistemas. Un servicio en la nube de alto nivel 304, tal como el servicio de computación distribuida de múltiples inquilinos de la figura 1, que incluye la funcionalidad descrita anteriormente de acuerdo con la figura 2, puede iniciar un proceso, tarea y/o trabajo (colectivamente "proceso") 306 en un servidor de aplicaciones mediante el envío de una llamada apropiada, o comando, 308 al servidor, que puede iniciar un hilo de trabajo 312 correspondiente, que representa una serie de instrucciones ejecutadas por el servidor para realizar el proceso 306 deseado. El proceso 306 puede estar asociado y/o especifico con un lenguaje de programación particular y el hilo de trabajo 312, por lo tanto, puede ejecutar una llamada 313 para instanciar un entorno de ejecución 316 a través de un motor de intérprete para continuar con la ejecución del proceso 306. En consecuencia, el proceso 306, a través del entorno de ejecución 316, puede realizar múltiples llamadas 318, 319 a un proxy de interfaz de objetos 320, tal como el proxy de interfaz de objetos 248 descrito anteriormente con referencia a la figura 2. Cada llamada representa una instrucción para realizar una unidad discreta de trabajo en el rendimiento del proceso 306. Las llamadas 318 pueden, a su vez, transmitirse y/o dirigirse a uno o más objetos de negocio adecuados y/o especificados 324, o, en el caso de la llamada 319, hacer que el proxy de interfaz de objetos 320 se realice de conformidad con la funcionalidad de su módulo de tolerancia a fallos 325.

El proxy de interfaz de objetos 320 puede recibir una llamada de guardar el estado 319, dando instrucciones para activar su funcionalidad de "guardar el estado". En respuesta a la llamada de guardar el estado 319, el proxy de interfaz de objetos 320 puede serializar la pila de ejecución actual del entorno de ejecución y sus referencias (por ejemplo, como se mantiene mediante el motor de intérprete), y guardar los datos binarios en serie a un almacén de datos (no mostrado). En caso de que se experimente una interrupción y/o evento inesperado, este estado serializado se puede restaurar una vez que el servicio en la nube de alto nivel 304 puede encontrar un entorno de ejecución adecuado.

La figura 4 representa un proceso de ejemplo que debería haber resultado en las llamadas 418(a)-(f). Sin embargo, el proceso experimentó una interrupción 420 después de la llamada 418(c), que conduce a la corrupción del entorno de ejecución 426. Como la llamada 418(c) fue una llamada de guardar el estado, se guardó el estado del entorno de

ejecución.

La figura 4b representa el proceso interrumpido que se muestra en la figura 4a que se reanuda después de la interrupción. Como existe un estado guardado de entorno de ejecución, el proceso reanudado instruye el hilo de trabajo 312 a reinstanciar un entorno de ejecución 424. El proxy de interfaz de objetos 426 restaura el entorno de ejecución reinstanciado 425 al estado guardado 428 del entorno de ejecución interrumpido de la figura 4a. El proceso reanudado puede continuar como si no hubiera habido una interrupción, lo que resulta en las llamadas 418(d)-(f).

Las figuras 5a y 5b representan otro ejemplo. Como se muestra en la figura 5a, el proceso debería dar lugar a las llamadas 518(a)-(g); sin embargo, se interrumpe 520 después de la llamada 518(f). La llamada de estado guardada más reciente era la llamada 518(c). Como se muestra en la figura 5b, como existe un estado guardado de entorno de ejecución, el proceso reanudado instruye el hilo de trabajo 312 a reinstanciar el entorno de ejecución 525. El proxy de interfaz de objetos 526 restaura el entorno de ejecución reinstanciado 425 al estado guardado 528 del entorno de ejecución interrumpido de la figura 5a. El proceso reanudado a continuación, continúa desde el punto de restauración, dando lugar a las llamadas 518(d)-(g).

Las figuras 4a-b y 5a-b muestran una llamada de guardar el estado que se hace explícitamente mediante el proceso de ejecución. Por ejemplo, una secuencia de comandos ejecutada por el motor de intérprete puede incluir instrucciones de lenguaje de secuencia de comandos que causan la llamada de guardar el estado. Alternativamente, o, además, el proxy de interfaz de objetos puede guardar automáticamente el estado de entorno de ejecución basado al menos en parte mediante la monitorización independientemente de las llamadas causadas por el entorno de ejecución.

La figura 6 representa un ejemplo de acuerdo con los presentes métodos y sistemas en los que el proxy de interfaz de objetos 604 dirige una serie de llamadas 606, 607 que pueden ser llamadas relativas solo a operaciones de lectura de datos, por ejemplo, las llamadas 606, o las llamadas relativas a las operaciones de escritura de datos, por ejemplo, las llamadas 607, desde el entorno de ejecución 608 para apropiarse de objetos de negocio 612. El proxy de interfaz de los objetos puede monitorizar en consecuencia las llamadas 606, 607 y de forma independiente realizar una operación de guardar el estado 616 para guardar el estado del entorno de ejecución 608 siguiendo cada uno de un tipo particular de llamada, tal como las llamadas 607 relativas a las operaciones de escritura de datos. El proxy de interfaz de objetos 604 guarda las decisiones de estado sobre la base de cualquier atributo adecuado de una llamada monitorizada y/o un conjunto monitorizado de llamadas, incluyendo el tipo de llamadas, el volumen de llamadas, la frecuencia de llamadas, el patrón de llamadas y los parámetros de llamadas. Este estado de guardado automático puede ofrecer una mayor integridad de los datos, menor mantenimiento de codificación y/o un nivel más limpio de abstracción a un coste de algo de rendimiento.

La figura 7 representa etapas 700 de ejemplo para permitir la tolerancia a fallos en un servicio en la nube de alto nivel, tal como el servicio de computación distribuida de múltiples inquilinos de la figura 1, de acuerdo con realizaciones ilustrativas de los presentes métodos y sistemas, como se muestra y se describe en relación con la figura 2. Para un proceso particular, un entorno de ejecución de aplicaciones de inquilinos puede ser instanciada 704. Por ejemplo, el intérprete puede instanciar un entorno de ejecución para una secuencia de comandos tal como se describe anteriormente. El intérprete puede luego ejecutar la secuencia de comandos 708 en el contexto del entorno de ejecución instanciado. La secuencia de comandos puede hacer una o varias llamadas de objeto de negocio 712 a un proxy de interfaz de objetos y los mensajes se puede encaminar 716 al objeto de negocio apropiado por el proxy.

Si el proxy de interfaz de objetos está configurado para crear puntos de restauración automática para el entorno de ejecución, como se describe con respecto a la figura 6, después de recibir una llamada 712, el proxy de interfaz de objetos puede probar 720 el progreso del proceso para detectar una condición de guardar el estado. Si se detecta una condición de guardar el estado 724, el procedimiento puede progresar a guardar 726 (por ejemplo, persistiendo) el estado actual del entorno de ejecución. La operación de guardar el estado puede realizarse antes de encaminar la llamada que la provocó, o en paralelo con la misma. Si una condición de guardar el estado no se detecta 728, el proxy de interfaz de objetos puede proceder a la recepción y al encaminamiento de la siguiente llamada de objeto de negocio. Si el proxy no está configurado para crear puntos de restauración automática, como se muestra en las figuras 4a-5b, (por ejemplo, por motivos de rendimiento), el estado entorno de ejecución se puede guardar 726 en respuesta a mensajes explícitos de "establecer punto de restauración". El proxy de interfaz de objetos entonces encamina la siguiente llamada al objeto de negocio adecuado y procesa la siguiente llamada 730.

Los diversos aspectos y realizaciones descritos anteriormente son ejemplos específicos, pero no exclusivos, de cómo se pueden implementar los presentes métodos y sistemas y las ventajas obtenidas de los mismos. Sin embargo, las personas que tienen experiencia ordinaria en la técnica reconocerán que las enseñanzas de los presentes métodos y sistemas son igualmente aplicables a otras realizaciones y/o se pueden describir de manera similar utilizando terminología alternativa. Por ejemplo, la descripción anterior del módulo de tolerancia a errores del proxy de interfaz de objetos se puede aplicar igualmente a cualquier proceso que se ejecuta en un sistema de computación distribuida que monitoriza, dirige, o sigue de otro modo la interacción entre varios otros elementos de

software que se ejecutan en el sistema, sigue el estado actual de uno o más de esos elementos, o su número, tipo, frecuencia, etc., y hace que se guarde el estado de uno o más de los elementos de software, o de otra manera crea una "copia de seguridad", en respuesta a una instrucción específica desde uno de los elementos de software o de acuerdo con un conjunto de reglas predefinidas. En el caso de un fallo u otra interrupción en la operación normal, el estado guardado más recientemente (o estados) puede restaurarse y la operación de los diversos elementos de software puede continuar a partir de ese punto, en lugar de tener que empezar de nuevo por completo.

A modo de un ejemplo no limitativo, la figura 8 representa aspectos de elementos que pueden estar presentes en una arquitectura de ordenador 800 a modo de ejemplo que puede estar configurada para implementar al menos algunas realizaciones de los presentes métodos y sistemas. La arquitectura 800 incluye subsistemas interconectados a través de un bus de sistema 802. Los subsistemas pueden incluir una impresora 804, un teclado 806, un disco fijo 808, y un monitor 810, que está acoplado a un adaptador de pantalla 812. Periféricos y dispositivos de entrada/salida (E/S), que se acoplan a un controlador E/S 814, se pueden conectar al sistema de ordenador mediante cualquier número de medios conocidos en la técnica, tales como un puerto bus serie universal (USB) 816. Por ejemplo, el puerto USB 816 o una interfaz externa 818 puede utilizarse para conectar el dispositivo de ordenador 800 a otros dispositivos y/o sistemas que no se muestran en la figura 8, que incluye una red de área amplia, tal como Internet, un dispositivo de entrada de ratón, y/o un escáner. La interconexión a través del bus del sistema 802 permite que uno o más procesadores 820 se comuniquen con cada subsistema y controlen la ejecución de instrucciones que pueden almacenarse en una memoria del sistema 822 y/o el disco fijo 808, así como el intercambio de información entre subsistemas. La memoria del sistema 822 y/o el disco fijo 808 pueden incorporar un medio tangible legible por ordenador.

Debe entenderse que los presentes métodos y sistemas como los descritos anteriormente se pueden implementar en forma de lógica de control utilizando el software de ordenador de una manera modular o integrada. Sobre la base de la divulgación y de las enseñanzas proporcionadas en el presente documento, una persona de experiencia ordinaria en la técnica conocerá y apreciará otras maneras y/o métodos para poner en práctica los presentes métodos y sistemas que utilizan hardware y una combinación de hardware y software.

Cualquiera de los componentes, procesos o funciones de software que se describen en esta solicitud pueden implementarse como código de software para ejecutarse mediante un procesador utilizando cualquier lenguaje de programación adecuado, tal como, por ejemplo, Java, C++, o Perl, utilizando, por ejemplo, técnicas orientadas a objetos o convencionales. El código de software puede almacenarse como una serie de instrucciones o comandos en un medio legible por ordenador, tal como una memoria de acceso aleatorio (RAM), una memoria de solo lectura (ROM), un medio magnético tal como una unidad de disco duro, un dispositivo de estado sólido, tal como una unidad de memoria flash, o un medio óptico tal como un CD-ROM. Cualquier medio legible por ordenador de este tipo puede residir en o dentro de un único aparato de cálculo, y puede estar presente en o dentro de diferentes aparatos de cálculo dentro de un sistema o red.

El uso de los términos "un" y "una" y "el" y referentes similares en la memoria descriptiva y en las reivindicaciones siguientes deben interpretarse para cubrir tanto el singular como el plural, a menos que se indique lo contrario en este documento o se contradiga claramente por el contexto. Los términos "que tiene", "que incluye", "que contiene" y referentes similares en la memoria descriptiva y en las reivindicaciones siguientes deben interpretarse como términos abiertos (por ejemplo, que significa "incluyendo, pero no limitado a") a menos que se indique lo contrario. La recitación de intervalos de valores en el presente documento está meramente pensada para servir como un método abreviado de referirse individualmente a cada valor separado que cae de manera inclusiva dentro del intervalo, a menos que se indique lo contrario en este documento, y cada valor separado se incorpora en la memoria descriptiva como si fuera recitado individualmente en este documento. Todos los métodos descritos en este documento se pueden realizar en cualquier orden adecuado, a menos que se indique lo contrario en este documento o se contradiga claramente en el contexto. El uso de cualquiera y todos los ejemplos, o lenguaje a modo de ejemplo (por ejemplo, "tal como") proporcionado en este documento, se pretende meramente para iluminar mejor las realizaciones de la invención y no plantea una limitación al alcance de la invención, a menos que se reivindique lo contrario. Ningún lenguaje en la memoria descriptiva debería interpretarse como una indicación de cualquier elemento no reivindicado como esencial para cada realización de la presente invención.

Las realizaciones ilustrativas de los presentes métodos y sistemas se han descrito en detalle anteriormente y en las figuras adjuntas para propósitos ilustrativos. Sin embargo, el alcance de los presentes métodos y sistemas se definen mediante las siguientes reivindicaciones y no se limitan a las realizaciones descritas anteriormente o representadas en las figuras. Las realizaciones que difieren de las descritas y que se muestran en este documento, pero aún dentro del alcance de los métodos y/o sistemas definidos se contemplan por los inventores y serán evidentes para las personas que tienen experiencia ordinaria en la técnica pertinente a la vista de esta memoria descriptiva en conjunto. Los inventores pretenden que los métodos y/o sistemas definidos se practiquen de manera distinta según se describe explícitamente en este documento.

REIVINDICACIONES

1. Un método para permitir una tolerancia a fallos en un proceso de cliente (306) que se ejecuta en un sistema de computación distribuida (200), comprendiendo el método:

(a) instanciar un entorno de ejecución (316) en respuesta a una llamada de un proceso de cliente (306), estando el entorno de ejecución (316) relacionado con el proceso de cliente (306) que se ejecuta en un sistema de computación distribuida (200);
(b) ejecutar instrucciones dentro de dicho entorno de ejecución (316), provocando dichas instrucciones que dicho entorno de ejecución (316) emita instrucciones adicionales a dicho sistema de computación distribuida (200), estando dichas instrucciones adicionales relacionadas con las acciones a realizar con respecto a los datos almacenados en dicho sistema de computación distribuida (200); y
(c) recibir dichas instrucciones adicionales mediante un proxy de interfaz de objetos (320), en donde al menos una de dichas instrucciones adicionales es una instrucción de guardar el estado, lo que hace que dicho proxy de interfaz de objetos (320) guarde un estado actual de dicho entorno de ejecución (316) en un almacén de datos.

2. Un método de la reivindicación 1, que comprende, además las etapas de:

(d) determinar que se ha producido un fallo con dicho proceso de cliente (306);
(e) instanciar un nuevo entorno de ejecución (420) en relación a dicho proceso de cliente (306);
(f) recibir una instrucción de restaurar el estado mediante un proxy de interfaz de objetos (320) que indica que dicho proceso de cliente (306) ha experimentado dicho fallo; y
(e) en respuesta a dicha instrucción de restaurar el estado, recuperar dicho estado actual de dicho almacén de datos y colocar dicho nuevo entorno de ejecución (426) en dicho estado actual.

3. Un método de la reivindicación 1 o la reivindicación 2, en el que dicho proxy de interfaz de objetos (320) funciona como una interfaz de programación de aplicaciones para dicho sistema de computación distribuida (200).

4. Un sistema para permitir una tolerancia a fallos en un proceso de cliente (306) que se ejecuta en un sistema de computación distribuida (200), que comprende:

(a) un primer entorno de ejecución (316) instanciado en un sistema de computación distribuida (200), el primer entorno de ejecución (316) instanciado en respuesta a una llamada de un proceso de cliente (306) y que ejecuta instrucciones en al menos un objeto de software (324) de acuerdo con el proceso de cliente (306);
(b) un proxy de interfaz de objetos (320) que se ejecuta en dicho sistema de computación distribuida (200) y que actúa como una interfaz para instrucciones de enrutamiento ejecutadas por dicho proceso de cliente (306) a dicho al menos un objeto de software (324) y que tiene un módulo de tolerancia a fallos (256); y
(c) un almacén de datos para almacenar datos relativos a un estado de dicho primer entorno de ejecución (316); y

en el que dicho módulo de tolerancia a fallos (256) está configurado para:

(i) recibir una instrucción de guardar el estado desde dicho primer entorno de ejecución (316), cuya recepción provoca que dicho módulo de tolerancia a fallos guarde un estado actual de dicho entorno de ejecución (316) en dicho almacén de datos.

5. Un sistema de la reivindicación 4, en el que dicho módulo de tolerancia a fallos (256) está configurado además para:

(ii) recibir una instrucción de restauración del estado desde un segundo entorno de ejecución (425), cuya recepción hace que dicho módulo de tolerancia a fallos (256) coloque el segundo entorno de ejecución (426) en dicho estado actual de dicho primer entorno de ejecución (316).

6. El sistema de la reivindicación 4 o la reivindicación 5, en el que dicho proxy de interfaz de objetos (320) funciona como una interfaz de programación de aplicaciones para dicho sistema de computación distribuida (200).

7. Un método de la reivindicación 2 o un sistema de la reivindicación 5, en donde cualesquiera instrucciones que se reciban con posterioridad a dicha instrucción de guardar el estado que se realizan mediante dicho sistema de computación distribuida (200), antes de determinar que dicho fallo se ha producido, se realizan de nuevo después de que dicho nuevo entorno de ejecución (426) se coloca en dicho estado actual.

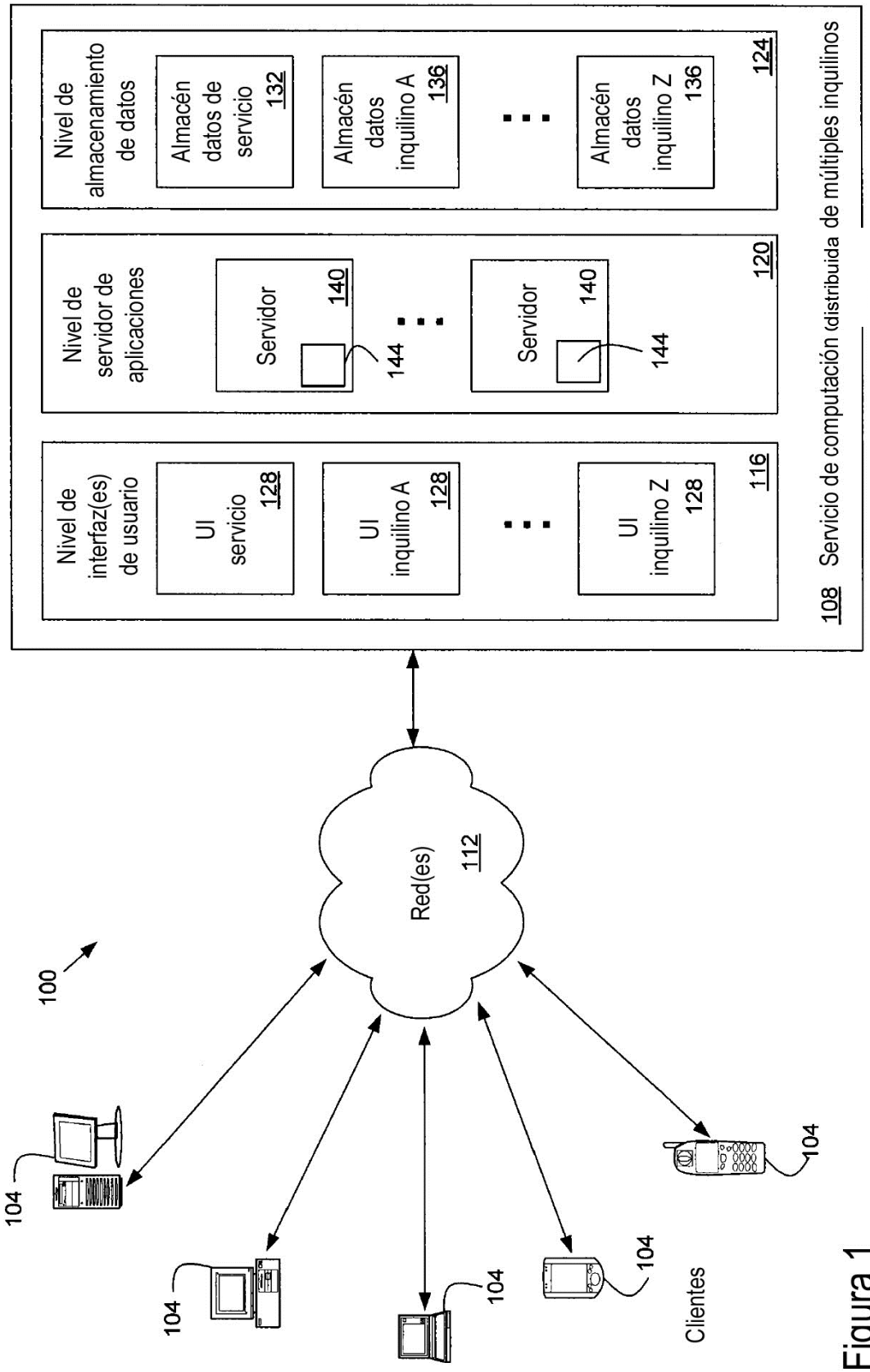
108 Servicio de computación distribuida de múltiples inquilinos
Nivel de interfaz(es) de usuario
UI servicio 128
UI inquilino A 128
UI inquilino Z 128
116
Nivel de servidor de aplicaciones
Servidor 140
144
Servidor 140
144
120
Nivel de almacenamiento de datos
Almacén datos de servicio 132
Almacén datos inquilino A 136
Almacén datos inquilino Z 136
124
Red(es) 112
100
104
104
104
104
Clientes

Figura 1

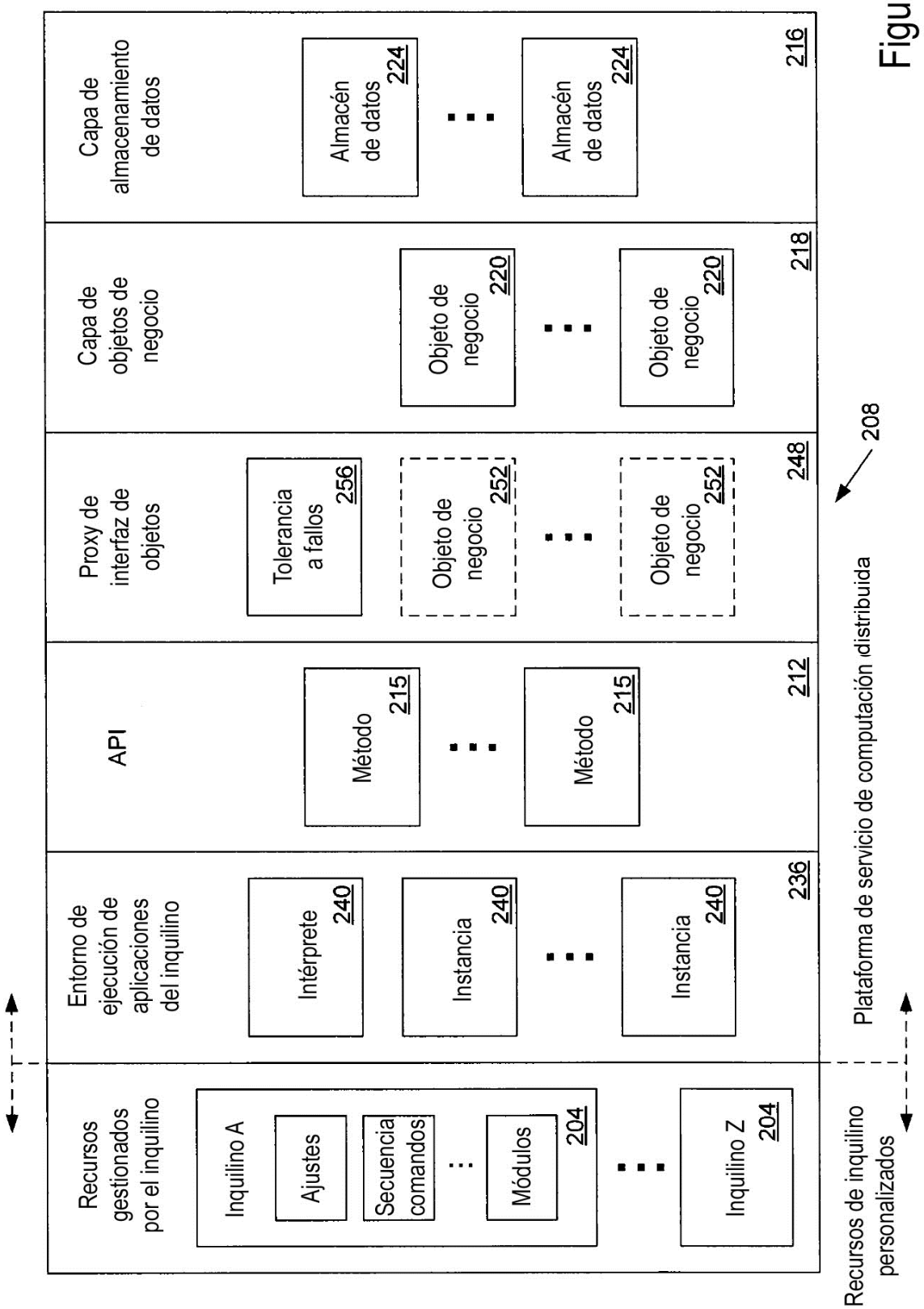
Recursos gestionados por el inquilino
Entorno de ejecución de aplicaciones del inquilino
API
Proxy de interfaz de objetos
Capa de objetos de negocio
Capa de almacenamiento de datos
Inquilino A
Ajustes
Secuencia comandos
Módulos
204
Inquilino Z
204
Intérprete
240
Instancia
240
Instancia
240
236
Método
215
Método
215
212
Tolerancia a fallos
256
Objeto de negocio
252
Objeto de negocio
252
248
Objeto de negocio
220
Objeto de negocio
220
218
Almacén de datos
224
Almacén de datos
224
216
208
Plataforma de servicio de computación distribuida
Recursos de inquilino personalizados

Figura 2

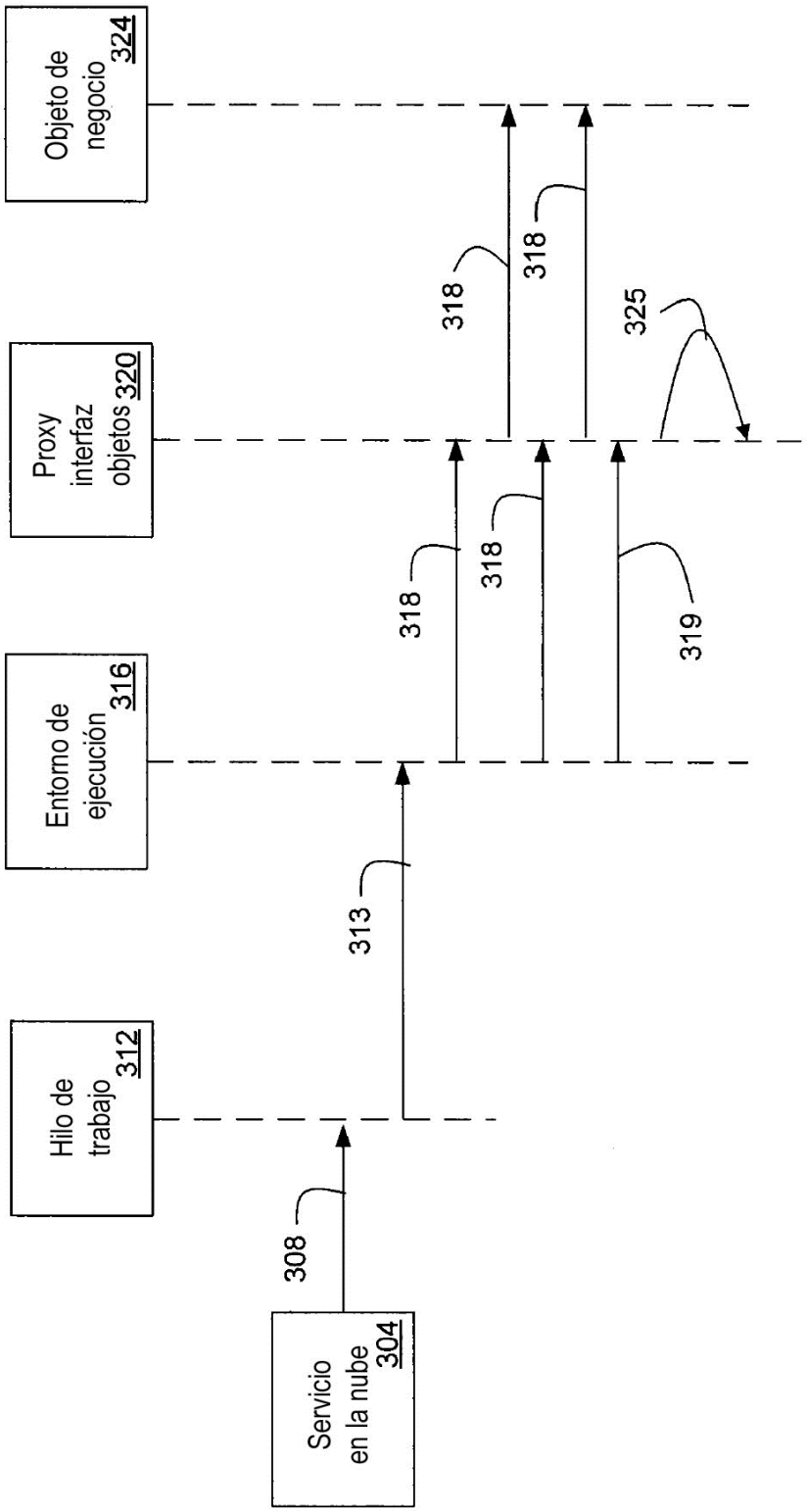
Objeto de negocio 324
Proxy interfaz objetos 320
Entorno de ejecución 316
Hilo de trabajo 312
Servicio en la nube 304
308
313
318
318
318
318
319
325

Figura 3

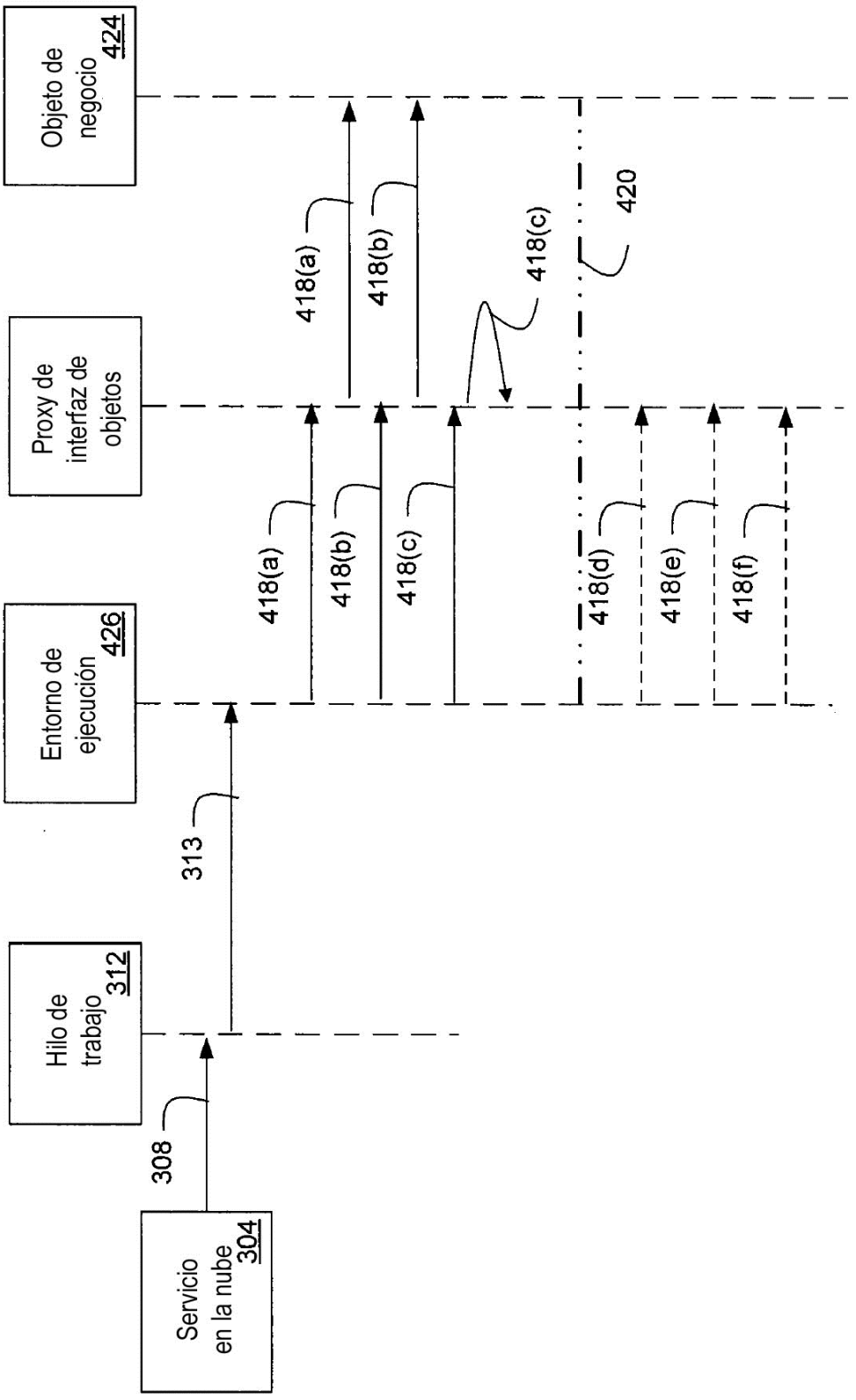
Servicio en la nube 304
Hilo de trabajo 312
Entorno de ejecución 426
Proxy de interfaz de objetos
Objeto de negocio 424
308
313
418(a)
418(b)
418(c)
418(d)
418(e)
418(f)
420

Figura 4a

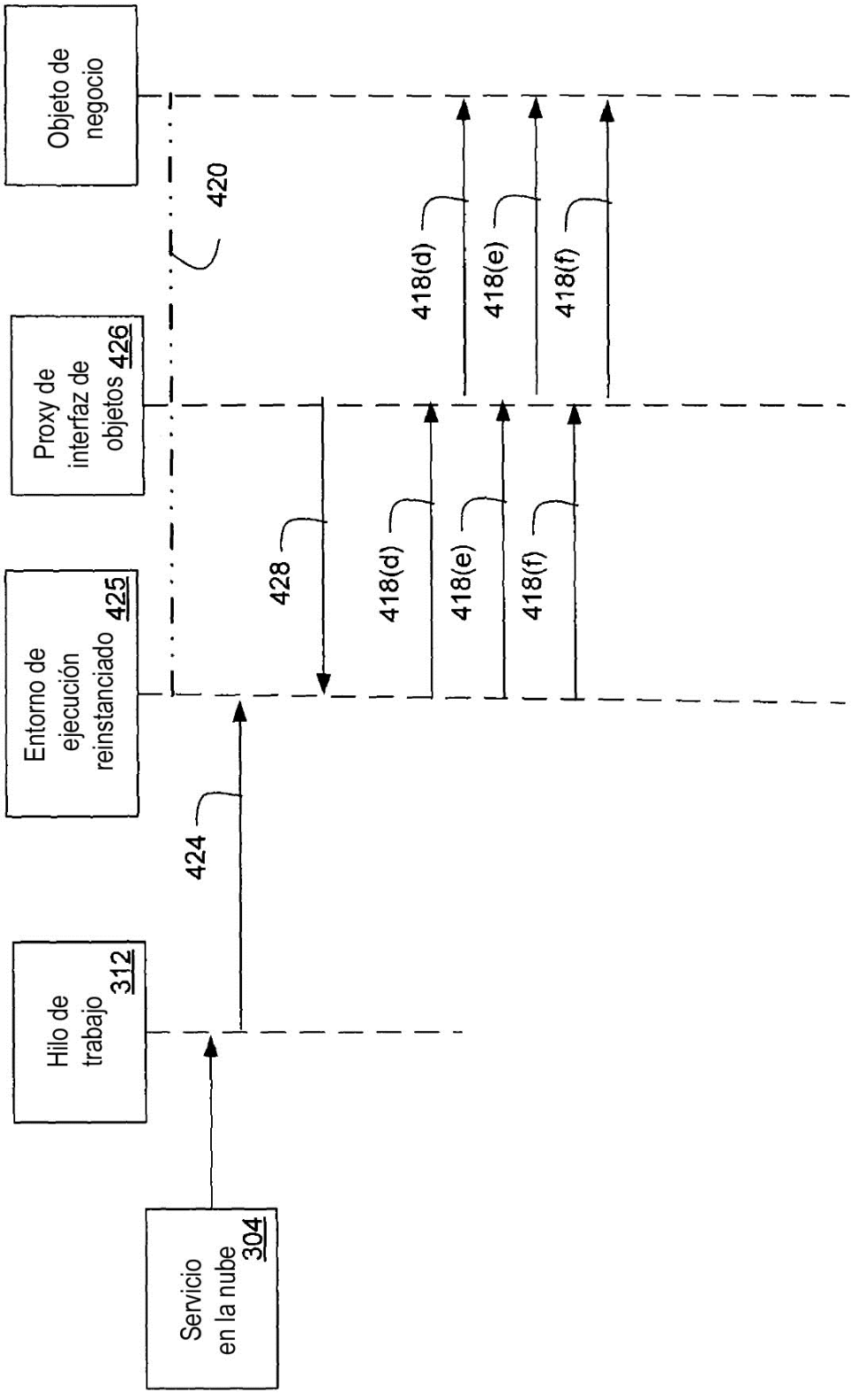
Servicio en la nube 304
Hilo de trabajo 312
Entorno de ejecución reinstanciado 425
Proxy de interfaz de objetos 426
Objeto de negocio
424
428
418(d)
418(e)
418(f)
420

Figura 4b

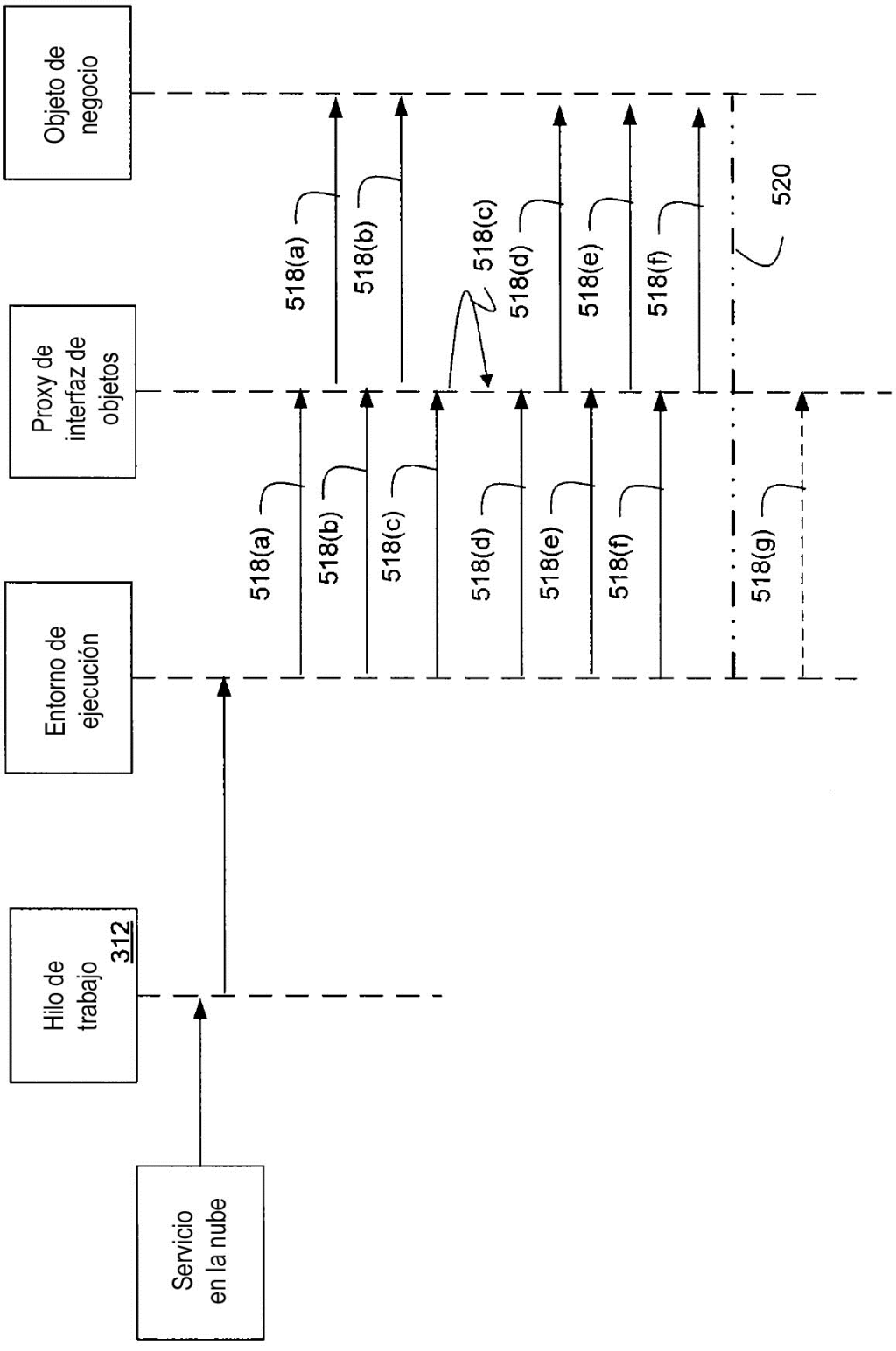
Objeto de negocio
Proxy de interfaz de objetos
Entorno de ejecución
Hilo de trabajo 312
Servicio en la nube
518(a)
518(b)
518(c)
518(d)
518(e)
518(f)
518(g)
520

Figura 5a

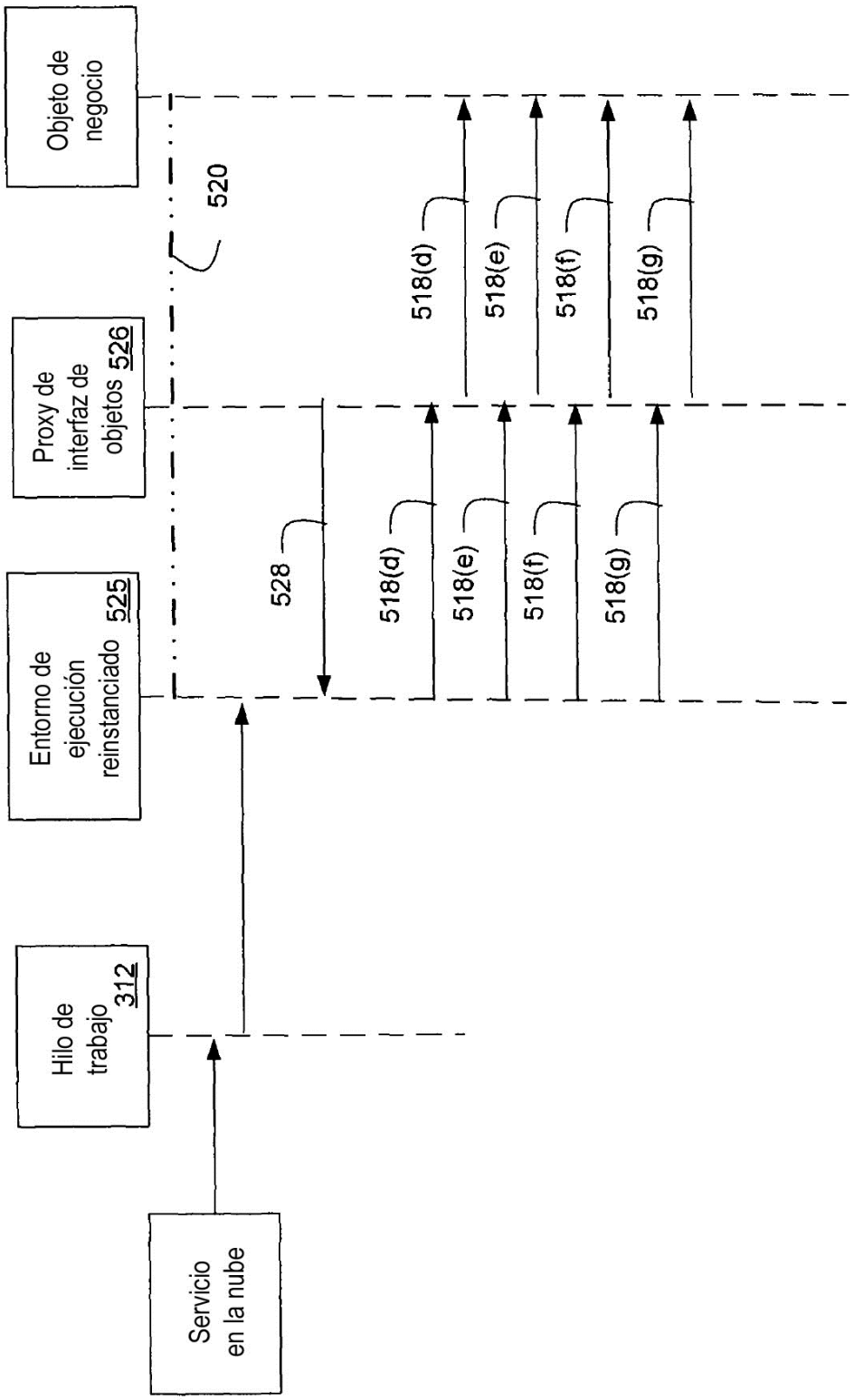
Objeto de negocio
520
Proxy de interfaz de objetos 526
Entorno de ejecución reinstanciado 525
Hilo de trabajo 312
Servicio en la nube
528
518(d)
518(e)
518(f)
518(g)
518(d)
518(e)
518(f)
518(g)

Figura 5b

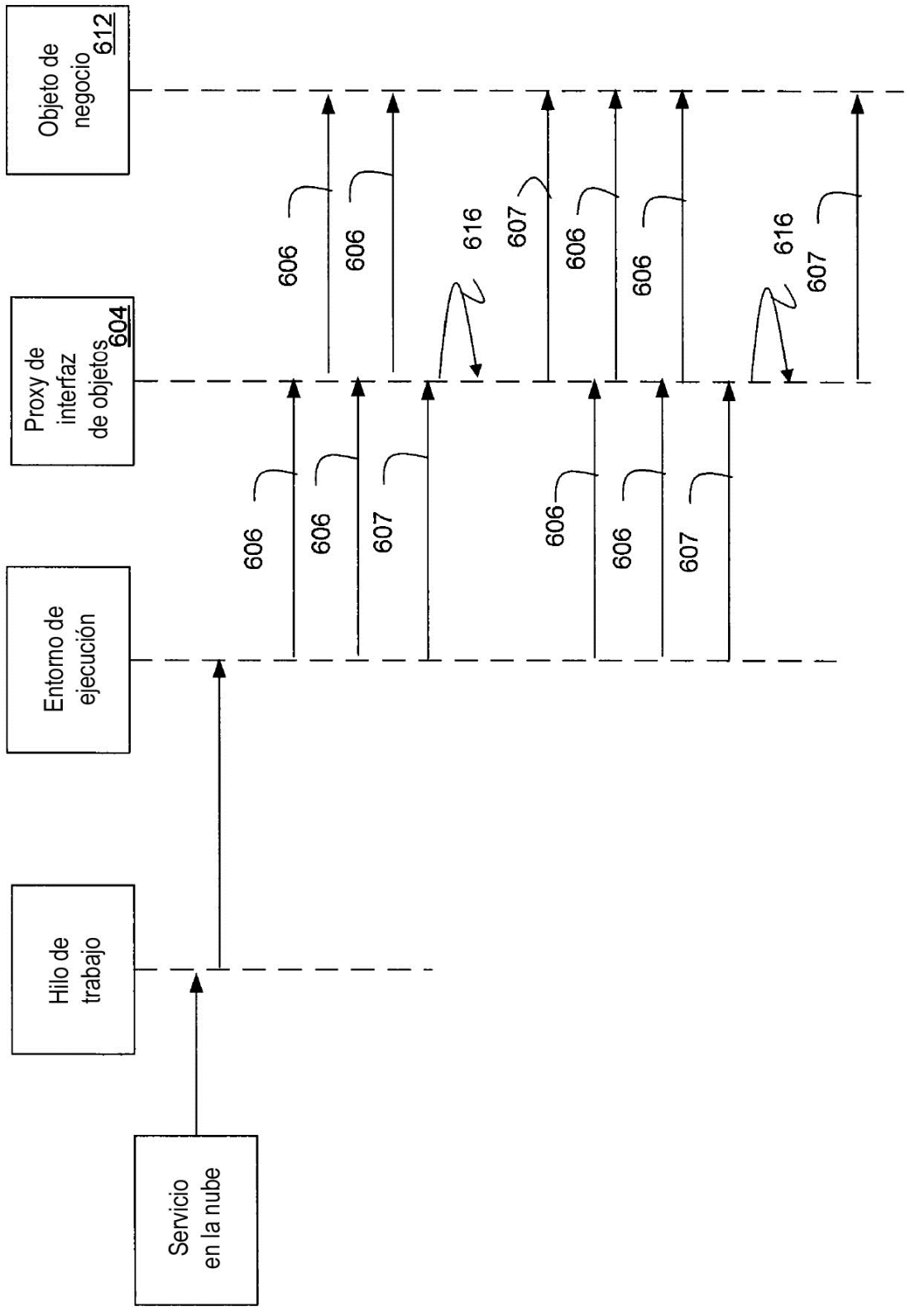
Servicio en la nube
Hilo de trabajo
Entorno de ejecución
Proxy de interfaz de objetos 604
Objeto de negocio 612
606
607
616

Figura 6

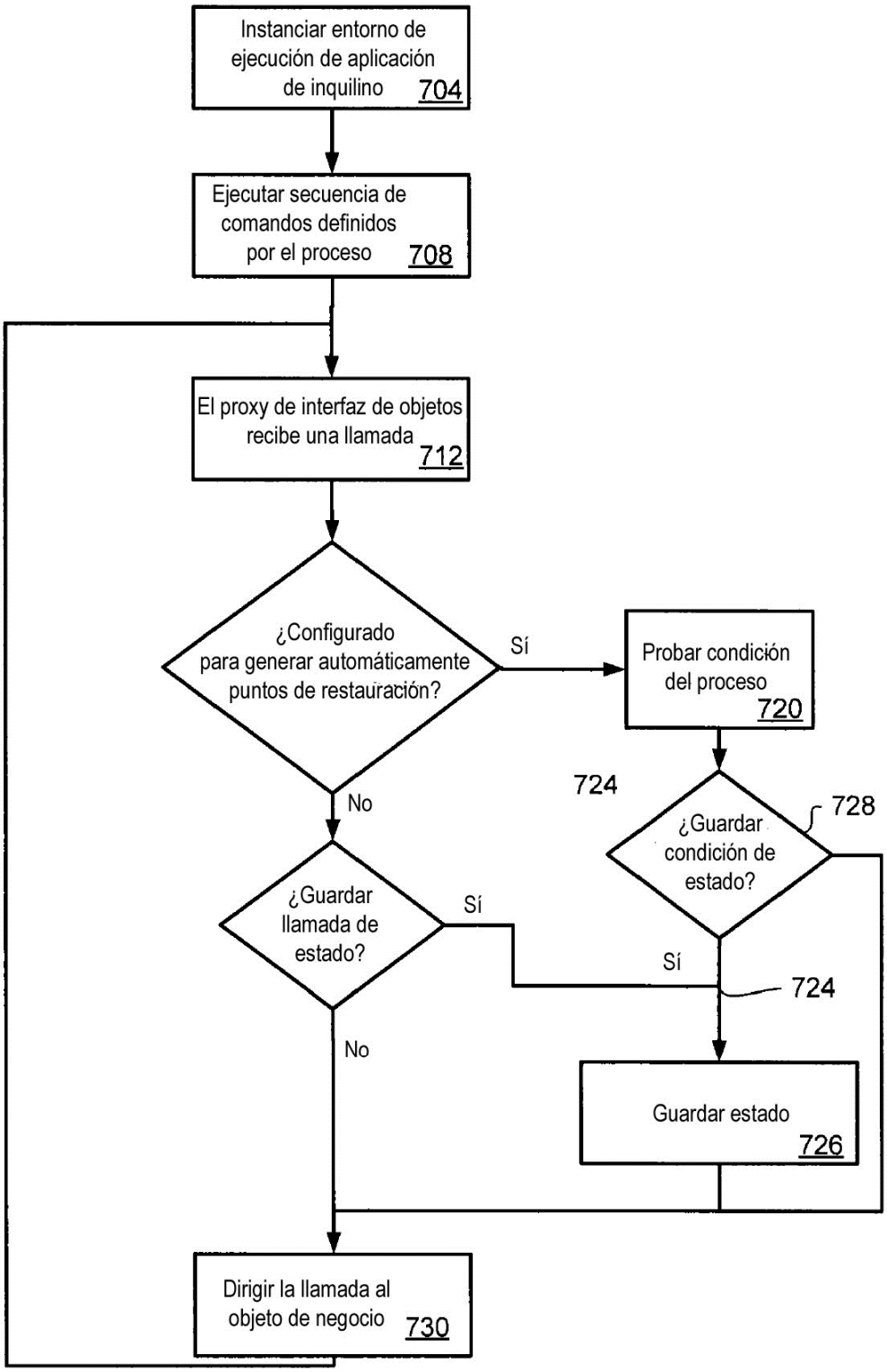
Instanciar entorno de ejecución de aplicación de inquilino 704
Ejecutar secuencia de comandos definidos por el proceso 708
El proxy de interfaz de objetos recibe una llamada 712
¿Configurado para generar automáticamente puntos de restauración?
Sí
Probar condición del proceso 720
724
¿Guardar condición de estado?
728
No
¿Guardar llamada de estado?
Sí
Sí
724
No
Guardar estado 726
Dirigir la llamada al objeto de negocio 730

Figura 7

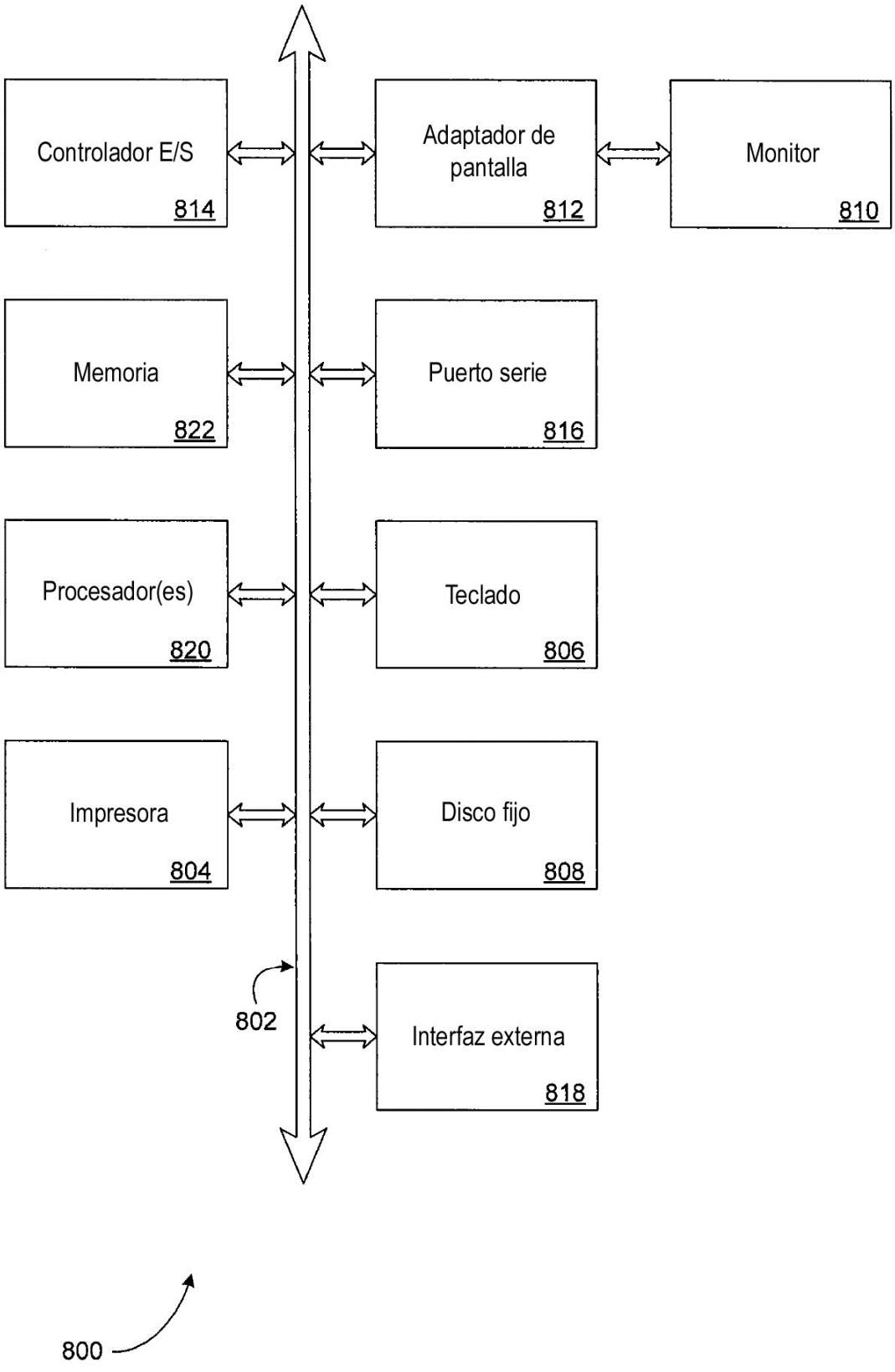
Controlador E/S
814
Adaptador de pantalla
812
Monitor
810
Memoria
822
Puerto serie
816
Procesador(es)
820
Teclado
806
Impresora
804
Disco fijo
808
802
Interfaz externa
818
800

Figura 8