(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 852 798**

(51) Int. CI.:

H04L 29/08 (2006.01)
G06F 15/16 (2006.01)
G06F 9/448 (2008.01)
G06F 8/60 (2008.01)
G06F 9/44 (2008.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **20.03.2018 PCT/CN2018/079583**

(87) Fecha y número de publicación internacional: **27.09.2018 WO18171578**

(96) Fecha de presentación y número de la solicitud europea: **20.03.2018 E 18772012 (3)**

(97) Fecha y número de publicación de la concesión europea: **16.12.2020 EP 3449618**

(54) Título: **Plataforma de nube sin servidor basada en gráficos de servicio**

(30) Prioridad:

20.03.2017 US 201762473973 P
16.03.2018 US 201815923989

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
14.09.2021

(73) Titular/es:

HUAWEI TECHNOLOGIES CO., LTD. (100.0%)
Huawei Administration Building, Bantian, Longgang District
Shenzhen, Guangdong 518129, CN

(72) Inventor/es:

ZHANG, HONG;
SUNAVALA, FARHAD P. y
FOURIE, HENRY LOUIS

(74) Agente/Representante:

PONS ARIÑO, Ángel

DESCRIPCIÓN

Plataforma de nube sin servidor basada en gráficos de servicio

Campo de la invención

La presente descripción se refiere a una arquitectura de nube sin servidor y, más específicamente, a la ejecución de funciones basadas en un gráfico de servicio dentro de la arquitectura de nube sin servidor.

ANTECEDENTES

La «nube» es una abstracción que se relaciona con la gestión de recursos en una red y, más específicamente, con una arquitectura de centro de datos que proporciona una plataforma para prestar servicios a través de una red. . Por ejemplo, la nube puede hacer referencia a diversos servicios prestados a través de Internet, como ser servicios de almacenamiento o servicios de proceso basados en red. Las implementaciones típicas de arquitectura de nube incluyen una jerarquía por capas que comprende una capa física de hardware de red, y una o más capas de software que permiten a los usuarios acceder al hardware de red. Por ejemplo, un tipo común de implementación de arquitectura de nube incluye una capa física de recursos de red (por ejemplo, servidores, matrices de dispositivos de almacenamiento, conmutadores de red, etc.) acompañada de un marco de software jerárquico de múltiples capas que incluye una primera capa que implementa la infraestructura como servicio (IaaS), una segunda capa que implementa la plataforma como servicio (PaaS) y una tercera capa que implementa el software como servicio (SaaS). En general, aunque puede haber excepciones, los recursos de la tercera capa dependen de los recursos de la segunda capa, los recursos de la segunda capa dependen de los recursos de la primera capa, y los recursos de la primera capa dependen de los recursos de la capa física.

Más recientemente, se ha desarrollado una arquitectura de nube sin servidor que permite a los usuarios ejecutar funciones en la nube sin aprovisionar recursos en la estructura jerárquica tradicional descrita anteriormente. Por ejemplo, Amazon Web Services (AWS) ha desarrollado un servicio conocido como Amazon® AWS Lambda que permite a los usuarios ejecutar código sin aprovisionar o administrar servidores para ejecutar el código, como en un servicio de proceso tradicional. En consecuencia, se puede desarrollar una aplicación web que llame a funciones cargadas en el servicio AWS Lambda, donde los recursos de proceso utilizados para ejecutar la función sean gestionados y aprovisionados por Amazon en lugar de por la aplicación web.

Sin embargo, este tipo de arquitectura sin servidor presenta algunos inconvenientes. Esta arquitectura es útil para ejecutar funciones específicas, que se pueden llamar varias veces desde varios clientes distintos, y los recursos para gestionar estas llamadas a funciones se aprovisionan automáticamente, lo que permite llevar a cabo un escalado para adaptarse al tráfico dinámico. No obstante, esta arquitectura no es adecuada para llamar a varias funciones seguidas. Recientemente, Amazon ha desarrollado otro servicio llamado AWS Step Functions que se puede utilizar con AWS Lambda para coordinar la ejecución de múltiples funciones. De hecho, AWS Step Functions permite a un usuario definir un flujo de trabajo para coordinar el orden y la ejecución de múltiples funciones. La ejecución de las funciones se puede retrasar hasta que se complete otra función, o incluso durante un período de tiempo establecido. Sin embargo, la ejecución de funciones habilitada por AWS Step Functions no puede ser desencadenada por eventos complejos y, por lo tanto, existe una limitación basada en la complejidad de las aplicaciones o procesos que se pueden crear mediante este servicio. Es necesario desarrollar una plataforma más diversa para ejecutar funciones en una arquitectura sin servidor similar a AWS Lambda.

El documento WO2017005329A1 describe un procedimiento para brindar un servicio (230) a un elemento de infraestructura solicitante (105, 110, 115; 245ie) perteneciente a una pluralidad de elementos de infraestructura (105, 110, 115 ; 245ie) interconectados para formar una red de datos (100).

El documento EP 0 662 651 A2 (AT & T CORP [EE. UU.]) de fecha 12 de julio de 1995 (12-07-1995) describe una máquina de estados en la cual las acciones también se pueden ejecutar cuando se reciben los eventos y no solo cuando se produce una transición al estado correspondiente, como sucede en las máquinas de estados comunes.

RESUMEN

La invención está definida por la arquitectura de nube sin servidor de la reivindicación independiente 1 y por el procedimiento de la reivindicación independiente 8. Las reivindicaciones dependientes definen realizaciones preferidas.

Breve descripción de los dibujos

Las Figuras **1A** y **1B** ilustran la infraestructura para implementar una nube, de acuerdo con la técnica anterior;

la Figura **2** es una ilustración conceptual de una arquitectura de nube, de acuerdo con la técnica anterior;

la Figura **3** es una ilustración conceptual de una arquitectura de nube sin servidor, de acuerdo con la técnica anterior;

la Figura **4A** es una ilustración conceptual de una arquitectura de nube sin servidor, de acuerdo con una realización;

la Figura **4B** es una ilustración conceptual de una arquitectura de nube sin servidor, de acuerdo con otra realización;

la Figura **5** es un diagrama conceptual de un modelo de máquina de estados, de acuerdo con una realización;

la Figura **6** ilustra componentes adicionales de la arquitectura de nube sin servidor de la Figura **4A,** de acuerdo con una realización;

la Figura **7** ilustra el funcionamiento del modelo de máquina de estados implementado por una instancia de máquina de estados, de acuerdo con una realización;

la Figura **8** es un diagrama de flujo de un procedimiento para ejecutar funciones alojadas en la nube en una arquitectura de nube sin servidor, de acuerdo con una realización; y

la Figura **9** ilustra un ejemplo de sistema en el que se pueden implementar las diversas arquitecturas y/o funciones de las diversas realizaciones anteriores.

Descripción detallada

Los gráficos de servicio organizan las funciones alojadas en la nube en una aplicación de microservicio coordinada. Los gráficos de servicio representan un modelo de máquina de estados impulsado por eventos de una amplia variedad de fuentes que controlan la ejecución de las funciones alojadas en la nube de una manera prescrita. Los gráficos de servicio permiten a un usuario organizar funciones alojadas en la nube para que se ejecuten secuencial o simultáneamente, gestionar las condiciones de error volviendo a invocar llamadas a función, gestionar el escalado para adaptarse a cargas de eventos variables, y acciones similares. Un controlador de gráfico de servicio permite la creación de instancias y el envío de múltiples instancias de máquina de estados, que implementan un modelo de máquina de estados según se define en un gráfico de servicio.

Los gráficos de servicio también permiten a un usuario definir puntos de encuentro (es decir, estados) para esperar eventos predefinidos antes de ejecutar una función alojada en la nube y avanzar por el gráfico de servicio . Las ventajas de coordinar la ejecución de funciones alojadas en la nube con gráficos de servicio es que los gráficos de servicio proporcionan un marco coherente para gestionar funciones alojadas en la nube sin que el usuario tenga que resolver los problemas de coordinación por su cuenta.

Las Figuras **1A** y **1B** ilustran la infraestructura para la implementación de una nube **100,** de acuerdo con la técnica anterior. La nube **100,** como se usa en este documento, se refiere al conjunto de recursos de hardware (de proceso, de almacenamiento y de red) ubicados en uno o más centros de datos (es decir, ubicaciones físicas) y al marco de software para implementar un conjunto de servicios en una red, tal como Internet. Como se muestra en la Figura **1A,** la nube **100** incluye una pluralidad de centros de datos **110,** incluyendo cada centro de datos **110** de la pluralidad de centros de datos **110** uno o más grupos de recursos **120.** Un grupo de recursos **120** incluye una capa de almacenamiento **122,** una capa de proceso **124** y una capa de red **126.**

Como se muestra en la Figura **1B**, la capa de almacenamiento **122** incluye los recursos físicos para almacenar instrucciones y/o datos en la nube **100.** La capa de almacenamiento **122** incluye una pluralidad de redes de área de almacenamiento (SAN) **152;** cada SAN **152** proporciona acceso a uno o más dispositivos de almacenamiento a nivel de bloque. En una realización, una SAN **152** incluye uno o más dispositivos de almacenamiento no volátiles accesibles a través de la red. Los ejemplos de dispositivos de almacenamiento no volátiles incluyen, pero no se limitan a, unidades de disco duro (HDD), unidades de estado sólido (SSD), memoria flash, tal como una EEPROM o una tarjeta Compact Flash (CF), y dispositivos similares. En otra realización, una SAN **152** es una matriz de almacenamiento RAID (matriz redundante de discos independientes) que combina múltiples componentes de unidad de disco físico (por ejemplo, varios HDD similares) en una única unidad lógica de almacenamiento. En otra realización más, una SAN **152** es un recurso de almacenamiento virtual que proporciona un nivel de abstracción a los recursos de almacenamiento físicos de forma tal que se pueda usar una dirección de bloque virtual para hacer referencia a datos almacenados en uno o más bloques de memoria correspondientes en uno o más dispositivos de almacenamiento físicos no volátiles. En tal realización, la capa de almacenamiento **122** puede incluir un marco de software, ejecutado en uno o más procesadores, para implementar los recursos de almacenamiento virtual.

La capa de proceso **124** incluye los recursos físicos para ejecutar procesos (es decir, conjuntos de instrucciones) en la nube **100**. La capa de proceso **124** puede incluir una pluralidad de unidades de escalado de procesos (CSU) **154**, incluyendo cada CSU **154** al menos un procesador y un marco de software para utilizar el al menos un procesador. En una realización, una CSU **154** incluye uno o más servidores (por ejemplo, servidores blade) que proporcionan hardware físico para ejecutar conjuntos de instrucciones. Cada servidor puede incluir uno o más procesadores (por ejemplo, CPU, GPU, ASIC, FPGA, DSP, etc.) así como memoria volátil para almacenar instrucciones y/o datos que han de ser procesados por el uno o más procesadores. La CSU **154** también puede incluir un sistema operativo, cargado en la memoria volátil y ejecutado por el uno o más procesadores, que proporciona un entorno en tiempo de ejecución para que se ejecuten diversos procesos en los recursos de hardware del servidor. En otra realización, una CSU **154** es una máquina virtual que proporciona una colección de recursos virtuales que emulan los recursos de hardware de un servidor. La capa de proceso **124** puede incluir un hipervisor o monitor de máquina virtual que permite que varias máquinas virtuales se ejecuten de manera sustancialmente simultánea en un único servidor.

La capa de red **126** incluye los recursos físicos para implementar redes. En una realización, la capa de red **126** incluye varios conmutadores y/o enrutadores que permiten que los datos se comuniquen entre los diferentes recursos de la nube **100**. Por ejemplo, cada servidor de la capa de proceso **124** puede incluir un controlador de interfaz de red (NIC) acoplado a una interfaz de red (por ejemplo, Ethernet). La interfaz puede estar acoplada a un conmutador de red que permite enviar datos desde ese servidor a otro servidor conectado al conmutador de red. La capa de red **126** puede implementar varias capas del modelo OSI, incluida la capa de vínculo de datos (es decir, la capa 2), la capa de red (es decir, la capa 3) y la capa de transporte (es decir, la capa 4). En una realización, la capa de red **126** implementa una capa de virtualización que permite que se establezcan redes virtuales dentro de la red física. En tales realizaciones, cada unidad de red (NU) **156** de la capa de red **126** es una red privada virtual (VPN).

Se deberá tener en cuenta que cada centro de datos **110** de la pluralidad de centros de datos puede incluir un conjunto diferente de recursos de hardware y, por lo tanto, una cantidad diferente de grupos de recursos **120.** Además, algunos grupos de recursos **120** pueden excluir una o más de: la capa de almacenamiento **122**, la capa de proceso **124**, y/o la capa de red **126**. Por ejemplo, un grupo de recursos **120** puede incluir solo un conjunto de servidores dentro de la capa de proceso **124.** Otro grupo de recursos **120** puede incluir una capa de proceso **124** y una capa de red **126,** pero ninguna capa de almacenamiento **122.**

La Figura **2** es una ilustración conceptual de una arquitectura de nube **200,** de acuerdo con la técnica anterior. Como se muestra en la Figura **2,** la arquitectura de nube **200** se representa como una pluralidad de capas jerárquicas. La arquitectura de nube **200** incluye una capa física **202,** una capa de infraestructura como servicio (IaaS) **204,** una capa de plataforma como servicio (PaaS) **206** y una capa de software como servicio (SaaS) **208.** La capa física **202** es la colección de recursos de hardware que implementan la nube. En una realización, la capa física **202** se implementa como se muestra en las Figuras **1A** y **1B.**

La capa de IaaS **204** es un marco de software que permite que los recursos de la capa física **202** se asignen a diferentes servicios de infraestructura. La capa de IaaS **204** puede incluir un marco de software para gestionar y asignar recursos en la capa física **202.** Por ejemplo, la capa de IaaS **204** puede incluir un motor de software para gestionar las SAN **152** en la capa de almacenamiento **122** de la capa física **202.** El motor de software también puede gestionar las CSU **154** en la capa de proceso **124** de la capa física **202.** Los servicios de la capa de PaaS **206** o de la capa de SaaS 208 pueden solicitar la asignación de servicios en la capa de IaaS **204** para ejecutar una tarea. Por ejemplo, un servicio de clúster de la capa de PaaS **206** puede solicitar la asignación de varias máquinas virtuales a la capa de IaaS **204**, que implementa las máquinas virtuales en una o más CSU **154.**

Se deberá tener en cuenta que la arquitectura de nube **200** mostrada en la Figura **2** es solo un tipo de marco de arquitectura implementado en nubes convencionales. Sin embargo, otras arquitecturas de nube pueden implementar marcos diferentes. Por ejemplo, una arquitectura de nube puede incluir la capa de IaaS **204** y la capa de SaaS **208** sin que intervenga ninguna capa de PaaS **206**. En otro ejemplo, una arquitectura de nube puede incluir una capa de contenedor como servicio (CaaS) (es decir, una nueva forma de virtualización de recursos sin IaaS ni PaaS) más una capa de SaaS encima de la capa de CaaS. En cada instancia, estas arquitecturas de nube emplean un esquema de dependencia de recursos para solicitar recursos en los que ejecutar el servicio.

La Figura **3** es una ilustración conceptual de una arquitectura de nube sin servidor **300,** de acuerdo con la técnica anterior. Como se muestra en la Figura **3,** la arquitectura de nube sin servidor **300** no tiene la misma estructura jerárquica que una arquitectura de nube tradicional, como se muestra en la Figura **2**. La capa física **302** es la colección de recursos de hardware que implementan la nube. En una realización, la capa física **302** se implementa como se muestra en las Figuras **1A** y **1B**.

La arquitectura de nube sin servidor **300** incluye un motor sin servidor **310** que gestiona la ejecución de funciones usando los recursos de hardware de la capa física **302**. En una realización, el motor sin servidor **310** incluye un hipervisor que gestiona una o más máquinas virtuales ejecutadas en recursos de la capa física **302**. El motor sin servidor **310** ejecuta software en cada máquina virtual que incluye uno o más contenedores para ejecutar diversas

funciones. El motor sin servidor **310** se configura para ejecutar las funciones basándose en llamadas a función recibidas desde una puerta de enlace de API **320**.

Una aplicación **330** puede llamar a una función haciendo una llamada a función. En una realización, una llamada a función se implementa haciendo una llamada a la API de REST a un punto de conexión asociado con la puerta de enlace de API **320.** Como se conoce en la técnica, los procedimientos estándar de protocolo de transferencia de hipertexto (HTTP) pueden usarse con un localizador uniforme de recursos (URL) para especificar una función identificada por el URL. La puerta de enlace de API **320** puede recibir llamadas a función desde la aplicación **330**, lo cual activa la ejecución de la función correspondiente por parte del motor sin servidor **310.**

Se deberá tener en cuenta que el término «sin servidor» no se refiere al hecho de que la arquitectura de nube **300** no incluye servidores, sino que el término «sin servidor» se refiere al hecho de que el llamador de la función no necesita aprovisionar recursos de servidor para ejecutar la función, ya que es el motor sin servidor **310** el que se ocupa de dicho aprovisionamiento. Además, se deberá tener en cuenta que el motor sin servidor **310** puede construirse sobre arquitecturas de nube convencionales de modo que el aprovisionamiento de máquinas virtuales, por ejemplo, utilice servicios convencionales de la capa de IaaS **204** o de la capa de PaaS **206**.

La Figura **4A** es una ilustración conceptual de una arquitectura de nube sin servidor **400,** de acuerdo con una realización. Como se muestra en la Figura **4A,** la arquitectura de nube sin servidor **400** no tiene la misma estructura jerárquica que una arquitectura de nube tradicional, como se muestra en la Figura **2**. La capa física **402** es la colección de recursos de hardware que implementan la nube. En una realización, la capa física **402** se implementa como se muestra en las Figuras **1A** y **1B.**

La arquitectura de nube sin servidor 400 se configura para permitir la ejecución de una pluralidad de funciones alojadas en la nube basadas en un modelo de máquina de estados que crea transiciones entre estados en respuesta a eventos. Una representación del modelo de máquina de estados puede estar basada en un lenguaje de gráfico de servicio. Por ejemplo, el modelo de máquina de estados puede definirse utilizando un gráfico de servicio, que es un archivo que incluye una representación del modelo de máquina de estados escrito en un lenguaje de gráfico de servicio. El modelo de máquina de estados comprende estados, acciones y eventos definidos en una estructura jerárquica. Las acciones pueden incluir la invocación de funciones, el procesamiento de carga útil, la espera durante un período de retraso, la transición al siguiente estado o la terminación de la máquina de estados. En una realización, el lenguaje de gráfico de servicio es una representación JSON de un modelo de máquina de estados. En otra realización, el lenguaje de gráfico de servicio es un lenguaje propio que tiene una sintaxis para definir el modelo de máquina de estados.

La arquitectura de nube sin servidor **400** incluye un motor de ejecución de funciones **410,** un motor de gráfico de servicio **420,** un administrador de gráfico de servicio (SG) **430,** y una o más fuentes de eventos **440**. El motor de ejecución de funciones **410** gestiona el aprovisionamiento de recursos en la capa física **402** para ejecutar funciones alojadas en la nube. Por ejemplo, el motor de ejecución de funciones **410** recibe llamadas a función del motor de gráfico de servicio **420,** aprovisiona un contenedor para gestionar la ejecución de la función o funciones, transfiere datos de carga útil al nodo en la nube asociado con el contenedor, ejecuta la función o funciones, y dirige los datos de carga útil resultantes a una ubicación de destino. Las funciones se pueden escribir en distintos lenguajes (por ejemplo, Java, Python, C++, etc.), que se compilan en ejecutables binarios o se compilan en tiempo de ejecución, y se ejecutan en un contenedor que aísla la función y los recursos asignados a la función de otras funciones.

El motor de gráfico de servicio **420** incluye un programador **422** de controlador de gráfico de servicio (SGC), uno o más controladores de gráfico de servicio (SGC) **424,** y una o más instancias de máquina de estados (SMI) **426** asociadas con cada SGC **424.** Cada SGC **424** se configura para gestionar una o más instancias de máquina de estados (SMI) **426,** que implementan un modelo de máquina de estados para una invocación de gráfico de servicio específica. Un gráfico de servicio definido por un usuario puede ser invocado por una llamada incluida en una aplicación web. Por ejemplo, una aplicación web puede incluir una solicitud HTTP asociada con un URI correspondiente al gráfico de servicio. En respuesta a la solicitud HTTP, se creará una nueva SMI **426** para implementar el modelo de máquina de estados definido por el gráfico de servicio para la aplicación web.

El programador **422** de SGC se configura para aprovisionar tantos SGC **424** como sean necesarios para gestionar el tráfico dinámico asociado con un gráfico de servicio específico (es decir, una colección de funciones asociadas con una aplicación web). Como ejemplo, un usuario crea una aplicación web que incluye una llamada al gráfico de servicio. El usuario carga la definición del gráfico de servicio en la nube, que se analiza para crear una especificación de modelo de máquina de estados. El motor de gráfico de servicio **420** se configura para aprovisionar y gestionar una serie de SMI **426** a medida que los clientes cargan y ejecutan la aplicación web en equipos cliente, ejecutando así código en el equipo cliente que incluye la llamada al gráfico de servicio. Cada SGC **424** y una serie de SMI **426** se pueden alojar en un nodo diferente (es decir, un servidor) en la nube. El programador **422** de SGC puede gestionar la creación de instancias de los SGC **424** en diversos nodos en respuesta al tráfico generado por la aplicación web, aumentando y reduciendo el número de SGC **424** según sea necesario. En una realización, el programador **422** de SGC para un gráfico de servicio específico se aprovisiona con un URI específico que permite a la aplicación web realizar una

llamada al programador **422** de SGC para ejecutar un microservicio de gráfico de servicio. El programador **422** de SGC crea una nueva SMI **426** para gestionar la llamada transmitiendo un mensaje a uno de los SGC **424**. El programador **422** de SGC también puede pasar una carga útil recibida desde la aplicación web a la SMI **426** a través del SGC **424** de modo que la SMI **426** pueda procesar la carga útil de acuerdo con el gráfico de servicio. Una vez que el gráfico de servicio ha completado el procesamiento de la carga útil, la SMI **426** transmite la carga útil resultante al programador **422** de SGC para ser transmitida de vuelta a la aplicación web que hizo la llamada, y la SMI **426** puede ser eliminada.

El administrador de SG **430** invoca uno o más motores de gráfico de servicio **420** en diversos nodos de la nube. Cada motor de gráfico de servicio **420** puede estar asociado con un gráfico de servicio diferente creado por una pluralidad de usuarios diferentes para una pluralidad de aplicaciones web diferentes. Además, el administrador de SG **430** puede invocar múltiples motores de gráfico de servicio **420** para un único gráfico de servicio con el fin de escalar la arquitectura de nube sin servidor para aplicaciones web extremadamente grandes. El administrador de SG **430** es un módulo lógico centralizado que equilibra cargas y realiza un escalado hacia adentro y hacia afuera de los distintos SGC **424** para cada gráfico de servicio. El administrador de SG **430** recibe gráficos de servicio a través de una API **450** que implementa una especificación de modelo de máquina de estados. El gráfico de servicio, escrito en un lenguaje de gráfico de servicio específico, puede ser leído por un analizador y convertido a una especificación de modelo de máquina de estados de acuerdo con la API **450** para implementar el modelo de máquina de estados a través de las SMI **426**.

El administrador de SG **430** también se configura para gestionar eventos. La nube puede incluir varias fuentes de eventos **440**. Las fuentes de eventos **440 hacen** referencia a cualquier componente de la nube que esté asociado con eventos. Los ejemplos de fuentes de eventos **440** incluyen, pero no se limitan a, dispositivos de almacenamiento en red, bases de datos, puertas de enlace de API y elementos similares. Los ejemplos de eventos incluyen, pero no se limitan a, un evento de archivo (por ejemplo, almacenar una imagen en un dispositivo de almacenamiento en la nube), un evento de tabla (por ejemplo, agregar una entrada a una base de datos) o un evento de protocolo (por ejemplo, recibir una solicitud HTTP en una puerta de enlace de API). Los eventos pueden usarse para desencadenar acciones en estados del modelo de máquina de estados. Dicho de otro modo, es posible que una acción no se ejecute inmediatamente al entrar en un estado específico, sino que se produzca únicamente en respuesta a uno o más eventos después de entrar en dicho estado.

En una realización, para monitorear los eventos, el administrador de SG **430** crea un agente de asignación de eventos **442** en cada fuente de eventos **440** a la que un gráfico de servicio hace referencia. El agente de asignación de eventos **442** es un módulo de software que se configura para recuperar una tabla de asignación de eventos a SGC y establecer un canal de comunicación entre la fuente de eventos **440** y uno o más SGC **424**. En una realización, el administrador de SG **430** genera una tabla de asignación que correlaciona eventos con SGC **424** en base a la especificación de modelo de máquina de estados correspondiente a un gráfico de servicio. Se utiliza una especificación de modelo de máquina de estados en particular para crear una instancia de (es decir, configurar) un motor de gráfico de servicio **420** específico, que incluye una serie de SGC **424.** Se puede hacer referencia a cada SGC **424** utilizando un identificador uniforme de recursos (URI) específico que permita que otros componentes de la arquitectura de nube sin servidor **400** se comuniquen directamente con los SGC **424**. Cada agente asignación de eventos **442** puede entonces establecer una conexión TCP/IP con uno o más SGC **424** usando los URI correspondientes a los SGC **424**. El agente de asignación de eventos **442** puede entonces ser configurado por el administrador de SG **430** para detectar uno o más eventos en una fuente de eventos **440** correspondiente. Después de detectar un evento, el agente de asignación de eventos **442** transmite entonces un mensaje directamente al uno o más SGC **424** correspondiente(s) al evento.

En una realización alternativa, el agente de asignación de eventos **442** puede ser un componente centralizado que sondea una pluralidad de fuentes de eventos **440** diferentes para detectar eventos. El agente de asignación de eventos **442** centralizado puede entonces transmitir mensajes relacionados con los eventos al uno o más SGC **424.**

Cada SMI **426** se configura para implementar una instancia del modelo de máquina de estados. La SMI **426** se invoca y pasa a un estado inicial. A continuación, la SMI **426** se ejecuta, procesando datos de carga útil mientras crea transiciones entre estados, según lo define el modelo de máquina de estados. Las acciones en cada estado pueden desencadenarse en respuesta a uno o más eventos. Las acciones pueden incluir invocar una llamada a función de una función alojada en la nube ejecutada por el motor de ejecución de funciones **410**. Las acciones también pueden desencadenarse cuando se reciben resultados de las funciones alojadas en la nube. Nuevamente, el modelo de máquina de estados implementado por la SMI **426** se usa para coordinar la ejecución de funciones alojadas en la nube en una aplicación de microservicio.

La Figura **4B** es una ilustración conceptual de una arquitectura de nube sin servidor **460,** de acuerdo con otra realización. Como opción, la arquitectura de nube sin servidor **460** puede implementarse con una o más características de una o más de cualquiera de las realizaciones expuestas en cualquiera de la(s) figura(s) anterior(es) y/o posterior(es) y/o en la descripción de la(s) misma(s). Por ejemplo, la arquitectura de nube sin servidor **460** puede implementarse en el contexto de la arquitectura de nube sin servidor **400** de la Figura **4A.** Sin embargo, se deberá tener en cuenta

que la arquitectura de nube sin servidor **460** puede implementarse en otros entornos adecuados.

Como se muestra, se proporciona un medio receptor en forma de un módulo receptor **462** para recibir una especificación de modelo de máquina de estados que define un modelo de máquina de estados. En diversas realizaciones, el módulo receptor **462** puede ser un componente, sin carácter restrictivo, del administrador de gráfico de servicio **430** de la Figura **4A,** al menos un procesador (que se describirá más adelante) y cualquier software que lo controle, y/o cualquier otro conjunto de circuitos capaz de realizar la funcionalidad antes mencionada.

También se incluye un medio generador de controladores en forma de un módulo generador de controladores **464** comunicado con el módulo receptor **462** para generar uno o más controladores de gráfico de servicio (SGC) configurados para gestionar instancias de máquina de estados (SMI) que implementan el modelo de máquina de estados. En diversas realizaciones, el módulo generador de controladores **464** puede ser un componente, sin carácter restrictivo, del administrador de gráfico de servicio **430** de la Figura **4A,** al menos un procesador (que se describirá más adelante) y cualquier software que lo controle, y/o cualquier otro conjunto de circuitos capaz de realizar la funcionalidad antes mencionada.

Continuando con la descripción de la Figura **4B,** el medio generador de agentes en forma de un módulo generador de agentes **466** está comunicado con el módulo generador de controladores **464** para generar una serie de agentes de asignación de eventos asociados con una o más fuentes de eventos, donde cada agente de asignación de eventos se configura para rastrear eventos generados en las fuentes de eventos y transmitir la notificación de los eventos al uno o más SGC. En diversas realizaciones, el módulo generador de agentes **466** puede ser un componente, sin carácter restrictivo, del administrador de gráfico de servicio **430** de la Figura **4A,** al menos un procesador (que se describirá más adelante) y cualquier software que lo controle, y/o cualquier otro conjunto de circuitos capaz de realizar la funcionalidad antes mencionada. También se incluyen medios de ejecución de funciones en forma de un módulo de ejecución de funciones **468** comunicado con el módulo generador de agentes **466** para ejecutar una pluralidad de funciones alojadas en la nube en uno o más nodos en respuesta a llamadas a función generadas por las SMI. En diversas realizaciones, el módulo de ejecución de funciones **468** puede ser un componente, sin carácter restrictivo, del motor de ejecución de funciones **410** de la Figura **4A,** al menos un procesador (que se describirá más adelante) y cualquier software que lo controle, y/o cualquier otro conjunto de circuitos capaz de realizar la funcionalidad antes mencionada.

La Figura **5** es un diagrama conceptual de un modelo de máquina de estados **500**, de acuerdo con una realización. Se deberá tener en cuenta que el modelo de máquina de estados **500** de la Figura **5** se proporciona solo como ejemplo, y los modelos de máquina de estados definidos por diversos usuarios pueden incluir una topología diferente de estados y transiciones entre estados. Como se muestra en la Figura **5**, el modelo de máquina de estados **500** incluye cinco estados: un primer estado **510**, un segundo estado **520**, un tercer estado **530**, un cuarto estado **540**, y un quinto estado **550.** El administrador de SG **430** recibe una llamada para invocar una instancia del modelo de máquina de estados **500**. El administrador de SG **430** puede crear agentes de asignación de eventos en una o más fuentes de eventos **440**, según sea necesario, basándose en cualquier evento asociado con los estados del modelo de máquina de estados **500**. En una realización, la llamada se puede realizar en respuesta a un equipo cliente que carga y ejecuta código del lado del cliente, como JavaScript, que genera una solicitud HTTP a un URL asociado con el modelo de máquina de estados **500**. El administrador de SG **430** también se configura para transmitir un mensaje al programador **422** de SGC para crear una nueva instancia de una SMI **426** correspondiente a la llamada. El programador **422** de SGC puede dirigir uno de los SGC **424** existentes o crear un nuevo SGC **424** para crear la nueva instancia de la SMI **426.** Ni bien se ejecuta la SMI 426 correspondiente a la llamada, la máquina de estados entra en el primer estado **510.**

Cada estado puede asociarse con una o más acciones. Las acciones pueden incluir llamar a una función basada en la nube, procesar una carga útil, retrasar una acción durante un período de tiempo, pasar al siguiente estado o terminar la máquina de estados. Las acciones se pueden invocar cuando se entra en un estado, cuando se han producido uno o más eventos, después de un retraso, cuando se recibe un resultado de una llamada a función, después de un error (por ejemplo, un tiempo de expiración de llamada a función) o al salir del estado. En muchos estados, una acción se invoca solo después de que ocurren uno o más eventos. Las acciones se pueden bloquear (es decir, su ejecución se puede bloquear) hasta que ocurran múltiples eventos (es decir, cuando se combinen con la lógica AND) o hasta que ocurra uno de dos o más eventos (es decir, cuando se combinen con la lógica OR). Nuevamente, la notificación de la aparición de eventos se recibe en un SGC **424** de uno o más agentes de asignación de eventos **442**.

Como se muestra en la Figura **5,** cuando se crea una SMI **426**, el modelo de máquina de estados **500** entra en un estado inicial, tal como el primer estado **510**. El primer estado **510** puede definir acciones que se ejecutan al entrar en el primer estado **510** o después de que hayan ocurrido uno o más eventos. El primer estado **510** también puede especificar condiciones para crear una transición a otro estado. En una realización, el modelo de máquina de estados puede pasar a otro estado cuando se devuelve un resultado de una función especificada por una acción invocada dentro del estado. En otra realización, el modelo de máquina de estados puede crear una transición a otro estado basándose en la aparición de uno o más eventos.

El modelo de máquina de estados **500** muestra una transición desde el primer estado **510** al segundo estado **520**. Sin embargo, un único estado también puede incluir lógica para dos o más transiciones a diferentes estados. Por ejemplo, se puede definir una primera transición desde el segundo estado **520** al tercer estado **530** en respuesta a la aparición de un primer evento, y se puede definir una segunda transición desde el segundo estado **520** a un cuarto estado **540** en respuesta a la aparición de un segundo evento. Dicho de otro modo, si el estado actual es el segundo estado **520,** entonces el siguiente estado del modelo de máquina de estados dependerá de la aparición de eventos específicos. Como se muestra en la Figura **5,** el tercer estado **530** incluye una primera transición al cuarto estado **540** y una segunda transición al quinto estado **550;** el cuarto estado **540** incluye una transición al quinto estado **550;** y el quinto estado **550** incluye una transición hasta la terminación del modelo de máquina de estados **500**.

Se deberá tener en cuenta que una SMI **426** se configura para implementar la lógica de un modelo de máquina de estados, llamando a funciones en respuesta a eventos, realizando transiciones entre estados, procesando cargas útiles recibidas de un cliente, uno o más eventos, un resultado de una acción, etc. En una realización, la SMI **426** es un objeto invocado usando una especificación de modelo de máquina de estados como entrada para configurar la SMI **426** para implementar el modelo de máquina de estados definido por la especificación de modelo de máquina de estados, que define los estados del modelo de máquina de estados, así como los eventos y acciones asociados con cada estado y las transiciones entre estados.

La Figura **6** ilustra componentes adicionales de la arquitectura de nube sin servidor **400** de la Figure **4A**, de acuerdo con una realización. Como se explicó anteriormente, el motor de gráfico de servicio **420** se configura en base a un modelo de máquina de estados representado por una representación de gráfico de servicio del modelo de máquina de estados. En una realización, la arquitectura de nube sin servidor **400** incluye un módulo de autenticación **610**, un módulo de análisis **620** y un módulo de validación **630**. Los usuarios pueden crear gráficos de servicios **602** que representan modelos de máquina de estados utilizando un lenguaje de gráfico de servicio. En una realización, el lenguaje de gráfico de servicio es una representación JSON de un gráfico de servicio. En otra realización, el lenguaje de gráfico de servicio es un lenguaje propio definido en una especificación de lenguaje de gráfico de servicio.

Un usuario puede enviar un gráfico de servicio **602** a un servicio en la nube que permite a un usuario crear modelos de máquina de estados para ejecutar funciones basadas en la nube en la arquitectura de nube sin servidor **400**. En una realización, el usuario almacena el gráfico de servicio **602** como un archivo y envía el archivo en una solicitud transmitida al servicio en la nube. El archivo que contiene el gráfico de servicio **602** puede ser procesado por un módulo de autenticación **610**. El módulo de autenticación **610** se configura para permitir que solo los usuarios autorizados creen nuevos gráficos de servicio **602** o modifiquen los gráficos de servicio **602** existentes. El módulo de autenticación **610** puede utilizar nombres de usuario y contraseñas para autenticar usuarios específicos y asegurarse de que el gráfico de servicio enviado corresponda a un usuario autorizado en particular. Una vez que el gráfico de servicio **602** ha sido autenticado, el gráfico de servicio **602** se pasa al módulo de análisis **620**.

El módulo de análisis **620** se configura para analizar el archivo, que puede recibirse en un formato de texto ASCII o similar, para determinar el modelo de máquina de estados representado por el gráfico de servicio **602**. En una realización, el módulo de análisis **620** analiza la sintaxis del gráfico de servicio para determinar la estructura del modelo de máquina de estados representado por el gráfico de servicio **602**. A medida que se lee cada objeto (es decir, estado, acción, evento y/o resultado) del archivo, los objetos se pueden pasar al módulo de validación **630**. El módulo de validación **630** se puede configurar para proporcionar validación de sintaxis de programación de los objetos de gráfico de servicio analizados. Dicho de otro modo, el módulo de validación **630** se configura para determinar si cada objeto especificado en el archivo para el gráfico de servicio **602** cumple con la sintaxis adecuada como se define en una especificación de lenguaje de gráfico de servicio. El módulo de validación **630** también puede validar cualquier función/evento al que se hace referencia en el gráfico de servicio **602**. Por ejemplo, el módulo de validación **630** puede verificar la existencia de cualquier función en el motor de ejecución de funciones **410** a la que se hace referencia mediante un nombre de función en el gráfico de servicio **602**. El módulo de validación **630** también puede verificar en el administrador de SG **430** la existencia de una fuente de eventos **440** asociada con cualquier referencia de evento mediante un nombre de evento en el gráfico de servicio **602**.

Una vez que el módulo de validación **630** ha confirmado la sintaxis y la disponibilidad de funciones y eventos, el módulo de validación **630** puede generar una especificación de modelo de máquina de estados **604** y una tabla de asignación de función a URI **606**. Nuevamente, la especificación de modelo de máquina de estados **604** es una definición del modelo de máquina de estados representado por el gráfico de servicio **602** y se transfiere como entrada cuando se crean instancias de las SMI **426** para implementar el modelo de máquina de estados. La tabla de asignación de función a URI **606** asigna nombres de funciones específicas (es decir, acciones) a un URI único para la función basada en la nube correspondiente al nombre de la función, de modo que la SMI **426** pueda generar una solicitud transmitida al motor de ejecución de funciones **410** usando el URI de la función en la tabla de asignación **606**.

La Figura **7** ilustra el funcionamiento del modelo de máquina de estados implementado por una instancia de máquina de estados, de acuerdo con una realización. Un gráfico de servicio puede verse como una colección de estados y transiciones entre esos estados. Cada estado puede desencadenar una o más acciones en respuesta a uno o más

eventos. La Figura **7** es un diagrama conceptual de las operaciones asociadas con entrar en un estado actual **710** en el modelo de máquina de estados implementado por una SMI **426**.

Como se muestra en la Figura **7,** el modelo de máquina de estados pasa de un estado anterior **705** al estado actual **710**. El estado anterior **705** transmite una carga útil como entrada al estado actual. La carga útil incluye datos que serán procesados por una o más funciones alojadas en la nube. En una realización, la carga útil son datos con formato de notación de objetos JavaScript (JSON). El estado actual **710** puede incluir un filtro de carga útil definido por el usuario **715**, que procesa la carga útil recibida del estado anterior **705**. El filtro de carga útil **715** puede incluir instrucciones para filtrar la entrada antes de que la entrada se proporcione como tal a las funciones alojadas en la nube. Se deberá tener en cuenta que el filtro de carga útil **715** es opcional y que la carga útil no necesita procesarse antes de pasarse a una acción **720**.

La acción **720** define cómo se ha de procesar la carga útil filtrada. La ejecución de la acción **720** puede ser retrasada por uno o más eventos **730**. De nuevo, un agente de asignación de eventos **442** monitorea las fuentes de eventos **440** y notifica al SGC **424** cuando ocurre un evento. En una realización, el agente de asignación de eventos **442** transmite una carga útil de evento al SGC **424,** que puede comprender datos con formato JSON relacionados con el evento. Por ejemplo, si el evento es un evento de archivo, entonces la carga útil del evento puede incluir metadatos con formato JSON asociados con el archivo, el contenido del archivo y elementos similares. Si el evento es un evento de tabla de base de datos, entonces la carga útil del evento puede incluir datos con formato JSON para una entrada en la tabla. Si el evento es un evento de protocolo, entonces la carga útil del evento puede incluir datos de protocolo con formato JSON, como un procedimiento HTTP y un URI asociado con el procedimiento.

La carga útil del evento puede ser procesada por un filtro de carga útil **735**, que puede procesar los datos de carga útil del evento antes de que los datos de carga útil del evento se pasen a la acción **720**. Una vez que se han cumplido las condiciones del evento, se procesa la acción **720**. En una realización, la acción **720** incluye invocar una llamada a función para una función alojada en la nube. La carga útil se puede pasar a la función **740** como entrada. La función **740** puede ser una función alojada en la nube ejecutada por el motor de ejecución de funciones **410**. El resultado de la función **740** se devuelve como una carga útil resultante. La carga útil resultante está formada por datos con formato JSON devueltos por la función. La carga útil resultante puede ser recibida por una acción resultante **750** en el estado. La acción resultante **750** puede desencadenar otra acción (es decir, acción anidada), que se puede retrasar en función de uno o más eventos adicionales, o puede desencadenar la transición del estado actual **710** al siguiente estado **760**. La carga útil resultante también se puede filtrar mediante un filtro de carga útil **755**, antes de que la carga útil resultante se transfiera al siguiente estado **760**.

La Figura **8** es un diagrama de flujo de un procedimiento **800** para ejecutar funciones alojadas en la nube en una arquitectura de nube sin servidor, de acuerdo con una realización. El procedimiento **800** puede llevarse a cabo con hardware, software o con una combinación de hardware y software. En una realización, el procedimiento **800** es implementado, al menos en parte, por el administrador de SG **430** de una arquitectura de nube sin servidor **400**.

En la etapa **802**, se recibe una especificación de modelo de máquina de estados que define un modelo de máquina de estados. La especificación de modelo de máquina de estados comprende un archivo que incluye una definición de un modelo de máquina de estados de acuerdo con una interfaz de programación de aplicaciones que define los componentes y la sintaxis para, a su vez, definir la estructura de un modelo de máquina de estados. En una realización, el administrador de SG **430** lee la especificación de modelo de máquina de estados de una base de datos que almacena una pluralidad de especificaciones de modelo de máquina de estados como las definen uno o más usuarios.

En la etapa **804**, se generan uno o más controladores de gráfico de servicio (SGC) configurados para gestionar instancias de máquina de estados (SMI) que implementan el modelo de máquina de estados. En una realización, el administrador de SG **430** crea al menos un SGC **424** en una memoria de un nodo de la nube. Se pueden implementar múltiples SGC **424** en múltiples nodos de la nube para equilibrar la carga de diferentes creaciones de instancias del gráfico de servicio por diversos equipos cliente. Cada SGC **424** puede crear una o más SMI **426** asociadas para implementar el modelo de máquina de estados.

En la etapa **806**, se generan varios agentes de asignación de eventos asociados con una o más fuentes de eventos. En una realización, el administrador de SG **430** genera un agente de asignación de eventos **442** para cada una de la una o más fuentes de eventos **440**. Cada agente de asignación de eventos **442** se configura para rastrear eventos generados en las fuentes de eventos **440** y transmitir notificaciones de los eventos a uno o más SGC **424**. El número de agentes de asignación de eventos **442** puede determinarse basándose en el número de fuentes de eventos a las que se hace referencia en el modelo de máquina de estados.

En la etapa **808**, se ejecutan una pluralidad de funciones alojadas en la nube en uno o más nodos en respuesta a llamadas a función generadas por las instancias de máquina de estados. En una realización, las SMI **426** se ejecutan y crean transiciones entre los estados del modelo de máquina de estados en respuesta a los eventos monitoreados por los agentes de asignación de eventos **442**. Las SMI **426** emiten llamadas a función invocadas como parte de las

acciones de uno o más estados, que llaman a las funciones alojadas en la nube ejecutadas por un motor de ejecución de funciones **410**.

La Figura **9** ilustra un ejemplo de sistema **900** en el que se pueden implementar las diversas arquitecturas y/o funcionalidades de las diversas realizaciones anteriores. Como se muestra, se proporciona un sistema **900** que incluye al menos un procesador **901** que está conectado a un bus de comunicación **902**. El bus de comunicación **902** puede implementarse utilizando cualquier protocolo adecuado, como PCI (interconexión de componentes periféricos), PCI-Express, AGP (puerto de gráficos acelerado), HyperTransport, o cualquier otro bus o protocolo(s) de comunicación punto a punto. El sistema **900** también incluye una memoria **904**. La lógica de control (software) y los datos se almacenan en la memoria **904** que puede adoptar la forma de memoria de acceso aleatorio (RAM).

El sistema **900** también incluye una interfaz de entrada/salida (E/S) **912** y una interfaz de comunicación **906**. La entrada del usuario puede recibirse desde los dispositivos de entrada **912**, por ejemplo, teclado, ratón, panel táctil, micrófono y elementos similares. En una realización, la interfaz de comunicación **906** puede acoplarse a un procesador de gráficos (no mostrado) que incluye una pluralidad de módulos de sombreado, un módulo de rasterización, etc. Cada uno de los módulos anteriores puede incluso estar situado en una única plataforma de semiconductores para formar una unidad de procesamiento de gráficos (GPU).

En la presente descripción, una plataforma única de semiconductores puede hacer referencia a un único circuito integrado o chip unitario basado en semiconductores. Cabe señalar que el término «plataforma única de semiconductores» también puede referirse a módulos de varios chips con mayor conectividad que simulan el funcionamiento en chip y realizan mejoras sustanciales con respecto a la utilización de una unidad central de procesamiento (CPU) convencional y a la implementación de buses. Desde luego, los diversos módulos también pueden estar situados por separado o en diversas combinaciones de plataformas de semiconductores según los deseos del usuario**.**

El sistema **900** también puede incluir un almacenamiento secundario **910**. El almacenamiento secundario **910** incluye, por ejemplo, una unidad de disco duro y/o una unidad de almacenamiento extraíble, que representa una unidad de disquete, una unidad de cinta magnética, una unidad de disco compacto, una unidad de disco versátil digital (DVD), un dispositivo de grabación, una memoria flash de bus serie universal (USB). La unidad de almacenamiento extraíble lee de, y/o escribe en, una unidad de almacenamiento extraíble de una manera muy conocida.

Los programas de ordenador, o algoritmos lógicos de control de ordenadores, se pueden almacenar en la memoria **904** y/o en el almacenamiento secundario **910**. Dichos programas informáticos, cuando se ejecutan, permiten al sistema **900** realizar diversas funciones. La memoria **904**, el almacenamiento **910** y/o cualquier otro almacenamiento son posibles ejemplos de medios legibles por ordenador.

En una realización, la arquitectura y/o funcionalidad de las diversas figuras anteriores pueden implementarse en el contexto del procesador **901**, un procesador de gráficos acoplado a la interfaz de comunicación **906**, un circuito integrado (no mostrado) que tenga al menos una parte de las funciones tanto del procesador **901** como de un procesador de gráficos, un conjunto de chips (es decir, un grupo de circuitos integrados diseñados para funcionar y vendidos como una unidad para realizar funciones relacionadas, etc.), y/o cualquier otro circuito integrado para el caso.

Aún así, la arquitectura y/o funcionalidad de las diversas figuras anteriores puede implementarse en el contexto de un sistema informático general, un sistema de placas de circuito, un sistema de consola de juegos concebido para entretenimiento, un sistema específico de aplicación y/o cualquier otro sistema deseado. Por ejemplo, el sistema **900** puede adoptar la forma de ordenador de sobremesa, ordenador portátil, servidor, estación de trabajo, consolas de juegos, sistema integrado y/o cualquier otro tipo de lógica. Aún así, el sistema **900** puede adoptar la forma de diversos dispositivos diferentes que incluyen, pero no se limitan a, un asistente digital personal (PDA), un teléfono móvil, un televisor, etc.

Además, aunque no se muestra, el sistema **900** puede estar acoplado a una red (por ejemplo, una red de telecomunicaciones, una red de área local (LAN), una red inalámbrica, una red de área extensa (WAN) tal como Internet, una red entre iguales, una red de cable, o redes similares) con fines de comunicación.

Se observa que las técnicas descritas en este documento, en un aspecto, se realizan en instrucciones ejecutables almacenadas en un medio legible por ordenador para ser usadas por, o en relación con, una máquina, aparato o dispositivo que ejecuta instrucciones, tal como una máquina, aparato o dispositivo informático o que contiene un procesador. Los expertos en la materia comprenderán que, para algunas realizaciones, se incluyen otros tipos de medios legibles por ordenador que pueden almacenar datos a los que puede acceder un ordenador, tales como casetes magnéticos, tarjetas de memoria flash, discos de vídeo digital, cartuchos Bernoulli, memoria de acceso aleatorio (RAM), memoria de solo lectura (ROM), y medios similares**.**

Como se usa en este documento, un «medio legible por ordenador» incluye uno o más de cualquier medio adecuado

para almacenar las instrucciones ejecutables de un programa informático de modo que el sistema, aparato, dispositivo o máquina que ejecuta instrucciones pueda leer (o recuperar) las instrucciones del medio legible por ordenador y ejecutar las instrucciones para llevar a cabo los procedimientos descritos. Los formatos de almacenamiento adecuados incluyen uno o más de los formatos electrónico, magnético, óptico y electromagnético. Una lista no exhaustiva de ejemplos de medios legibles por ordenador convencionales incluye: un disquete de ordenador portátil; una RAM; una ROM; una memoria de solo lectura programable y borrable (EPROM o memoria flash); dispositivos de almacenamiento óptico, incluidos un disco compacto (CD) portátil, un disco de vídeo digital (DVD) portátil, un DVD de alta definición (HD-DVD™), un disco BLU-RAY; y elementos similares.

Debe entenderse que la disposición de los componentes ilustrados en las figuras descritas se ofrece a modo de ejemplo y que también son posibles otras disposiciones. También debe entenderse que los diversos componentes (y medios) de sistema definidos por las reivindicaciones, descritos a continuación, e ilustrados en los diversos diagramas de bloque, representan componentes lógicos en algunos sistemas configurados de acuerdo con el tema descrito en este documento.

Por ejemplo, uno o más de estos componentes (y medios) de sistema se pueden realizar, en su totalidad o en parte, mediante al menos algunos de los componentes ilustrados en las disposiciones representadas en las figuras descritas. Además, si bien al menos uno de estos componentes se implementa al menos parcialmente como un componente de hardware electrónico y, por lo tanto, constituye una máquina, los otros componentes se pueden implementar en un software que, cuando se incluye en un entorno de ejecución, constituye una máquina, hardware o una combinación de software y hardware.

Más específicamente, al menos un componente definido por las reivindicaciones se implementa al menos parcialmente como un componente de hardware electrónico, tal como una máquina que ejecuta instrucciones (por ejemplo, una máquina basada en un procesador o que contiene un procesador) y/o como circuitos o conjunto de circuitos especializados (por ejemplo, puertas lógicas discretas interconectadas para realizar una función especializada). Otros componentes pueden implementarse en software, hardware o una combinación de software y hardware. Además, algunos o todos estos otros componentes se pueden combinar, algunos se pueden omitir por completo, y se pueden añadir componentes adicionales sin apartarse de la funcionalidad aquí descrita.

Por tanto, el tema que se describe en el presente documento se puede llevar a la práctica con muchas variaciones diferentes.

En la descripción anterior, el tema se describe con referencia a acciones y representaciones simbólicas de operaciones que son realizadas por uno o más dispositivos, a menos que se indique lo contrario. En este sentido, se entenderá que dichas acciones y operaciones, que en ocasiones se denominan ejecutadas por ordenador, incluyen la manipulación por parte del procesador de datos de forma estructurada. Esta manipulación transforma los datos o los mantiene en ubicaciones en el sistema de memoria del ordenador, lo que reconfigura o altera de otro modo el funcionamiento del dispositivo de una manera conocida para los expertos en la materia. Los datos se mantienen en ubicaciones físicas de la memoria como estructuras de datos que tienen propiedades específicas definidas por el formato de los datos. Sin embargo, aunque el tema se describe en el contexto anterior, no pretende ser limitante ya que los expertos en la materia comprenderán que las diversas acciones y operaciones descritas a continuación también se pueden implementar en hardware.

Para facilitar la comprensión del tema descrito en el presente documento, muchos aspectos se describen en términos de secuencias de acciones. Al menos uno de estos aspectos definidos por las reivindicaciones es realizado por un componente de hardware electrónico. Por ejemplo, se reconocerá que las diversas acciones pueden ser realizadas por circuitos o conjunto de circuitos especializados, por instrucciones de programa ejecutadas por uno o más procesadores, o por una combinación de ambos. La descripción de cualquier secuencia de acciones en el presente documento no pretende implicar que deba seguirse el orden específico descrito para llevar a cabo dicha secuencia. Todos los procedimientos descritos en este documento se pueden llevar a cabo en cualquier orden adecuado a menos que se indique lo contrario en este documento o que, por el contexto, sea claramente contradictorio.

El uso de los términos «un», «una», «el/la» y referentes similares usados en el contexto de la descripción del tema (especialmente en el contexto de las reivindicaciones incluidas a continuación) comprende tanto el singular como el plural, a menos que se indique lo contrario en este documento o que, por el contexto, sea claramente contradictorio. La mención de intervalos de valores en el presente documento pretende simplemente servir como un método abreviado para hacer referencia individualmente a cada valor separado que se encuentre dentro del intervalo, a menos que se indique lo contrario en el presente documento, y cada valor separado se incorpora en la memoria descriptiva como si se mencionara individualmente en el presente documento. Además, la descripción precedente tiene fines únicamente ilustrativos, y no restrictivos, ya que el alcance de la protección buscada está definido por las reivindicaciones que se exponen a continuación y por cualquier equivalente autorizado de las mismas. Todos y cada uno de los ejemplos o las expresiones que introducen ejemplos (como ser, «tal como») que se proporcionan en este documento sirven simplemente para ilustrar mejor el tema y no representan una limitación del alcance de dicho tema

a menos que se reivindique lo contrario. El uso del término «basado/a en» y otras frases similares que indiquen una condición para producir un resultado, tanto en las reivindicaciones como en la descripción escrita, no pretende excluir ninguna otra condición que provoque ese resultado. Ninguna expresión de la memoria descriptiva deberá interpretarse como indicación de que cualquier elemento no reivindicado es esencial para la puesta en práctica de las realizaciones según se reivindican.

Las realizaciones descritas en este documento incluyen el uno o más modos conocidos por el autor de la invención para poner en práctica la materia reivindicada. Se deberá tener en cuenta que las variaciones de dichas realizaciones resultarán evidentes para los expertos en la materia tras la lectura de la descripción precedente. El autor de la invención espera que los expertos empleen dichas variaciones según corresponda, y es su intención que el tema se lleve a la práctica de otras maneras que no sean las estrictamente descritas en este documento. Por consiguiente, el presente tema reivindicado incluye todas las modificaciones y equivalentes del tema mencionado en las reivindicaciones adjuntas, de acuerdo con la legislación aplicable. Asimismo, está contemplada cualquier combinación de los elementos descritos anteriormente con todas sus posibles variaciones a menos que se indique lo contrario en este documento o que, por el contexto, sea claramente contradictorio.

REIVINDICACIONES

1.Una arquitectura de nube sin servidor (400), que comprende:
un administrador de gráfico de servicio, SG, (430) configurado para:
recibir una especificación de modelo de máquina de estados que define un modelo de máquina de estados,
generar uno o más controladores de gráfico de servicio, SGC, (424) configurados para gestionar instancias de máquina de estados, SMI, (426) que implementan el modelo de máquina de estados,
generar una serie de agentes de asignación de eventos asociados con una o más fuentes de eventos (440), donde cada agente de asignación de eventos (422) está configurado para rastrear eventos generados en las fuentes de eventos (440), y transmitir la notificación de los eventos a uno o más SGC (424), y
gestionar eventos, donde los eventos se utilizan para desencadenar acciones en los estados del modelo de máquina de estados; y
un motor de ejecución de funciones (410) configurado para ejecutar una pluralidad de funciones alojadas en la nube en uno o más nodos en respuesta a llamadas a función generadas por las SMI (426);
donde las acciones incluyen invocar una llamada a función de una función alojada en la nube ejecutada por el motor de ejecución de funciones.

2. La arquitectura de nube sin servidor (400) de la reivindicación 1, donde una representación del modelo de máquina de estados está basada en un lenguaje de gráfico de servicio, donde el modelo de máquina de estados comprende estados, acciones y eventos definidos en una estructura jerárquica.

3. La arquitectura de nube sin servidor (400) de la reivindicación 2, donde los eventos incluyen al menos uno de: eventos de archivo, eventos de tabla y eventos de protocolo.

4. La arquitectura de nube sin servidor (400) de la reivindicación 2, donde el lenguaje de gráfico de servicio comprende la representación con notación de objetos JavaScript, JSON, del modelo de máquina de estados.

5. La arquitectura de nube sin servidor (400) de la reivindicación 1, donde el administrador de gráfico de servicio (430) se configura para generar un motor de gráfico de servicio (420) correspondiente a un gráfico de servicio específico, comprendiendo el motor de gráfico de servicio (420):
el uno o más SGC (424);
un programador de gráfico de servicio (422) comunicado con el uno o más SGC (424); y
una o más SMI (426) asociadas con cada SGC (424) en el uno o más SGC (424).

6. La arquitectura de nube sin servidor de la reivindicación 5, donde cada SGC (424) en el uno o más SGC (424) está alojado en un nodo diferente en una pluralidad de nodos, alojando cada nodo un SGC (424) y una o más SMI (426) asociada(s) con el SGC (424).

7. La arquitectura de nube sin servidor de la reivindicación 5, donde el administrador de SG (430) se configura para realizar equilibrio de cargas ajustando una cantidad del uno o más SGC (424).

8. Un procedimiento para ejecutar funciones alojadas en la nube en una arquitectura de nube sin servidor (400), comprendiendo el procedimiento:
recibir, en un administrador de gráfico de servicio, SG, (430), una especificación de modelo de máquina de estados que define un modelo de máquina de estados;
generar, en el administrador de SG, uno o más controladores de gráfico de servicio, SGC, (424) configurados para gestionar instancias de máquina de estados, SMI, (426) que implementan el modelo de máquina de estados;
generar, en el administrador de SG, una serie de agentes de asignación de eventos (442) asociados con una o más fuentes de eventos (440), donde cada agente de asignación de eventos (442) se configura para rastrear eventos generados en las fuentes de eventos (440), y transmitir la notificación de los eventos al uno o más SGC (424); y
gestionar, en el administrador de SG, eventos, donde los eventos se utilizan para desencadenar acciones en estados del modelo de máquina de estados; y
ejecutar, en un motor de ejecución de funciones, una pluralidad de funciones alojadas en la nube en uno o más nodos en respuesta a llamadas a función generadas por las SMI (426);
donde las acciones incluyen invocar una llamada a función de una función alojada en la nube ejecutada por el motor de ejecución de funciones.

9. El procedimiento de la reivindicación 8, donde una representación del modelo de máquina de estados está basada en un lenguaje de gráfico de servicio, donde el modelo de máquina de estados comprende estados, acciones y eventos definidos en una estructura jerárquica.

10. El procedimiento de la reivindicación 9, donde los eventos incluyen al menos uno de: eventos de archivo, eventos de tabla y eventos de protocolo.

11. El procedimiento de la reivindicación 9, donde el lenguaje de gráfico de servicio comprende la representación con notación de objetos JavaScript, JSON, del modelo de máquina de estados.

12. El procedimiento de la reivindicación 8, donde el administrador de gráfico de servicio se configura para generar un motor de gráfico de servicio (420) correspondiente a un gráfico de servicio específico, comprendiendo el motor de gráfico de servicio (420):
el uno o más SGC (424);
un programador (422) de gráfico de servicio comunicado con el uno o más SGC (424); y
una o más SMI (426) asociadas con cada SGC (424) en el uno o más SGC (424).

13. El procedimiento de la reivindicación 12, donde cada SGC (424) en el uno o más SGC (424) está alojado en un nodo diferente en una pluralidad de nodos, alojando cada nodo un SGC (424) y una o más SMI (426) asociada(s) con el SGC (424).

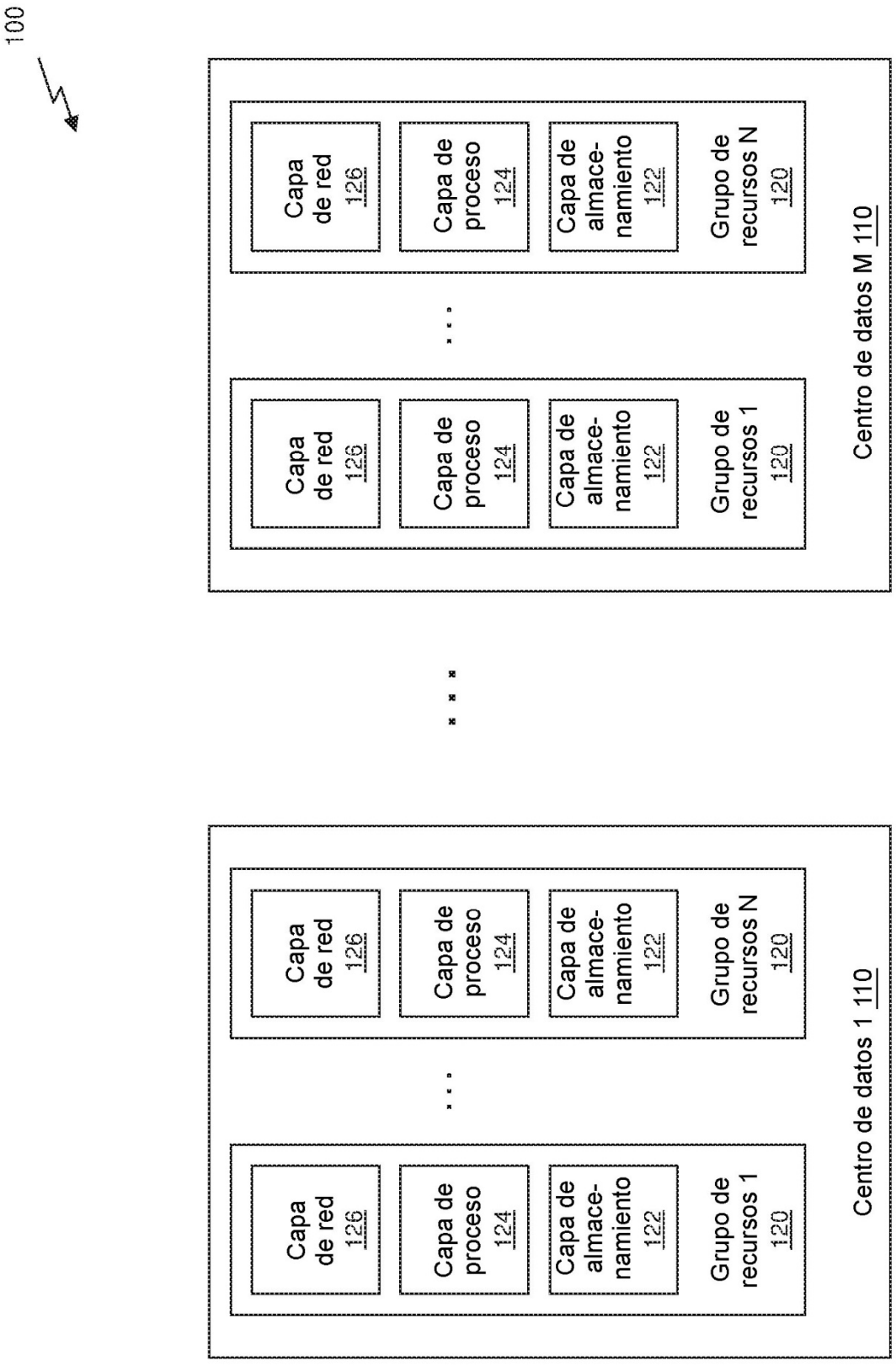

Fig 1A (técnica anterior)

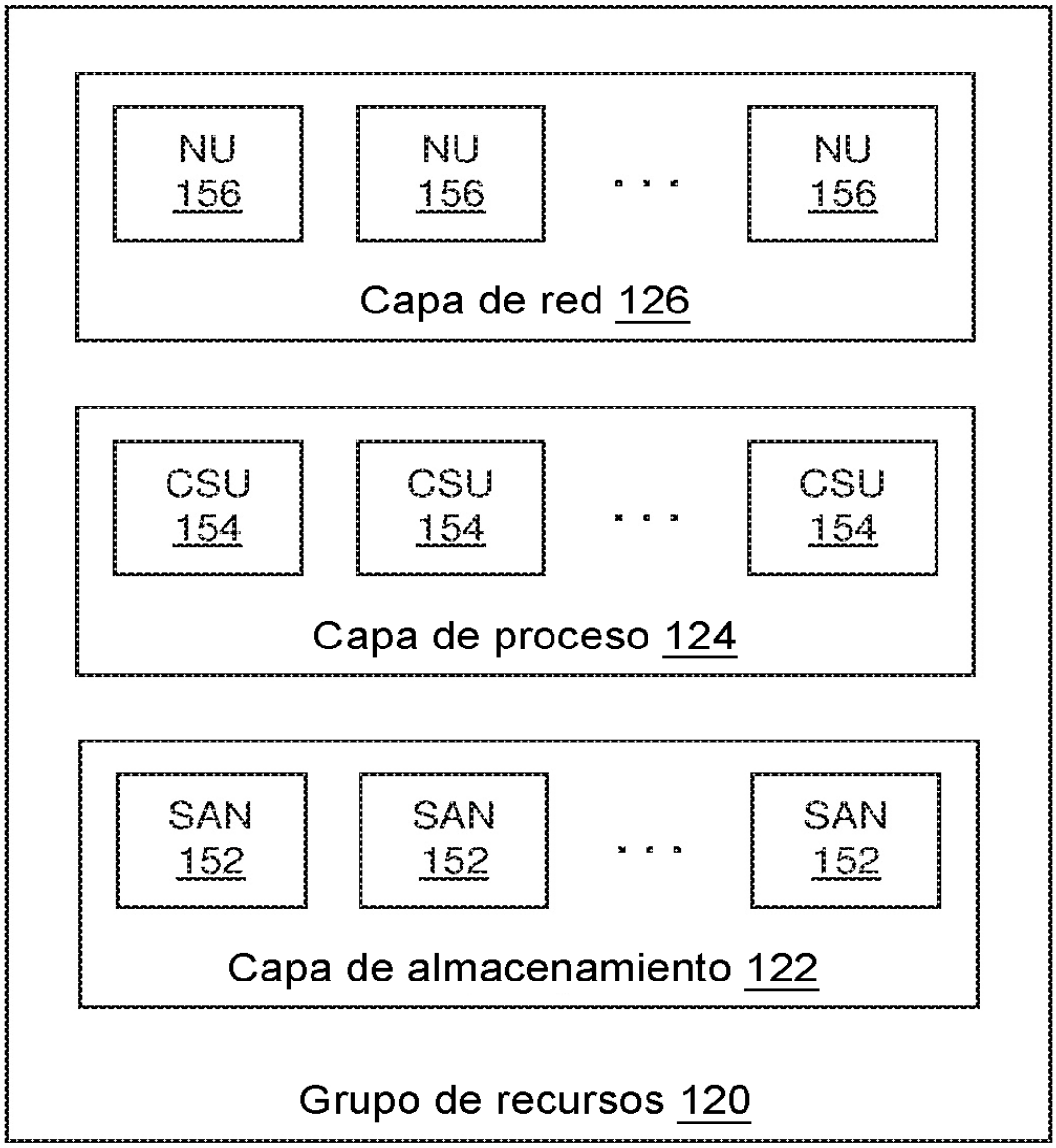

Fig. 1B (técnica anterior)

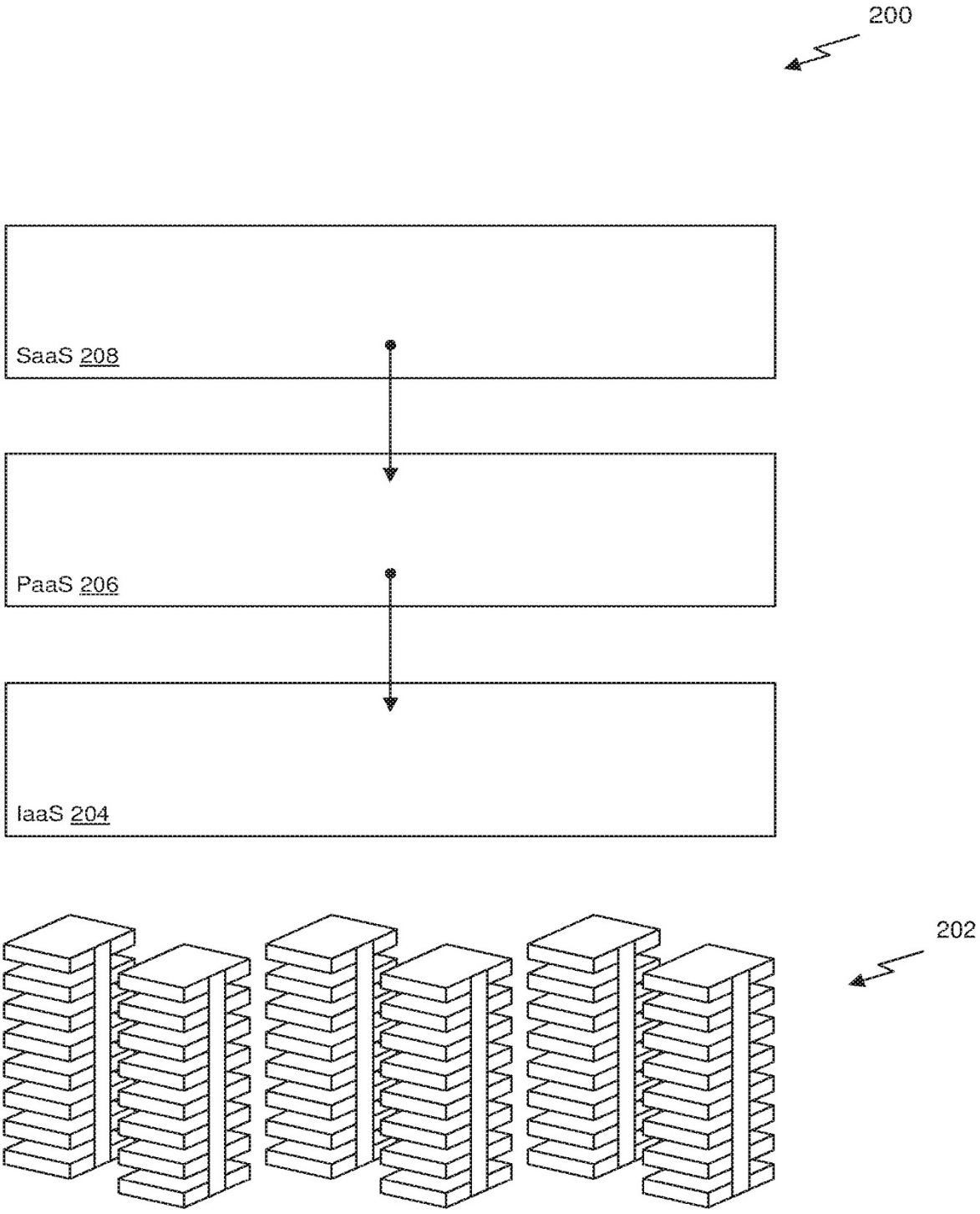

Fig. 2 (técnica anterior)

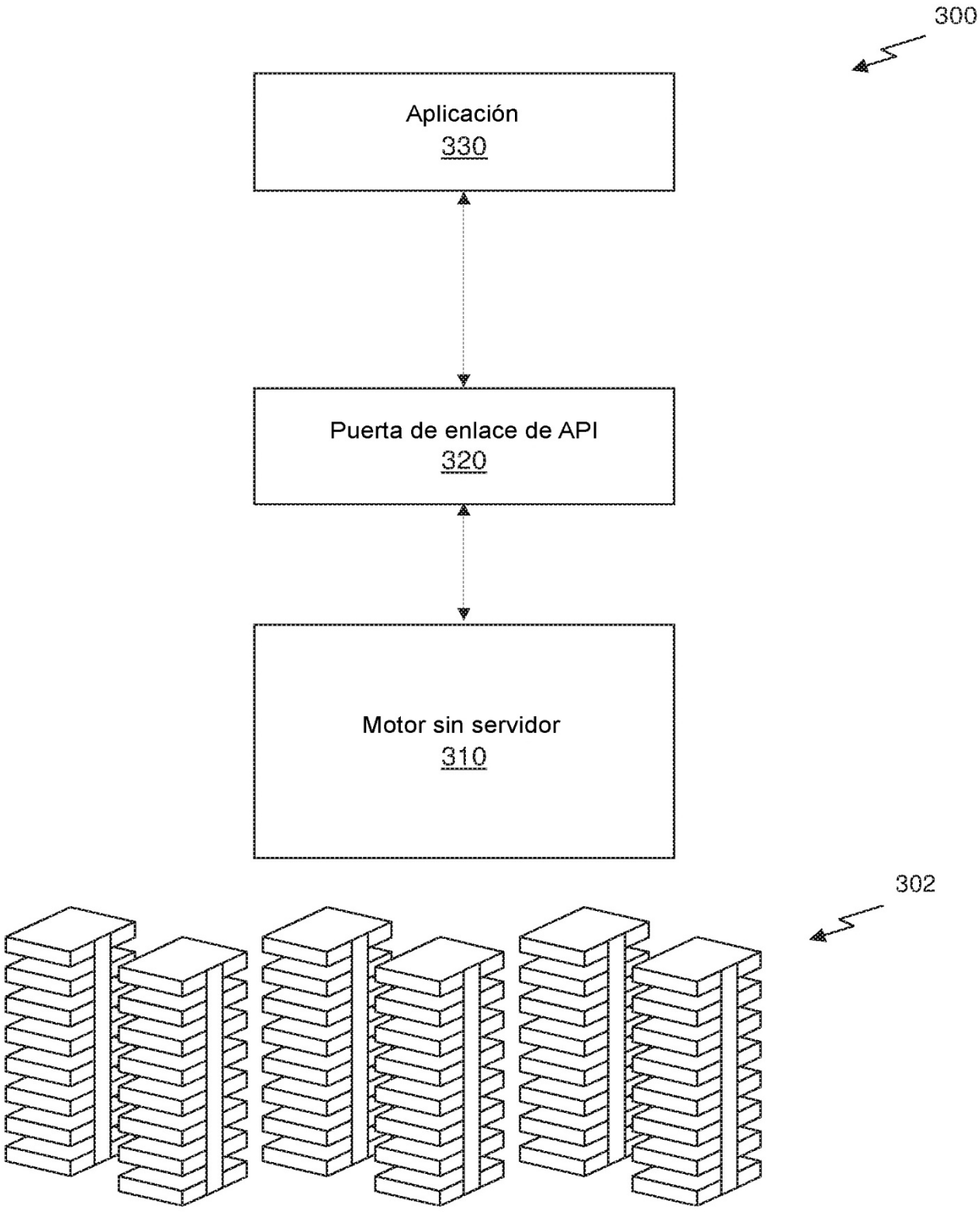
300
Aplicación
330
Puerta de enlace de API
320
Motor sin servidor
310
302

Fig. 3

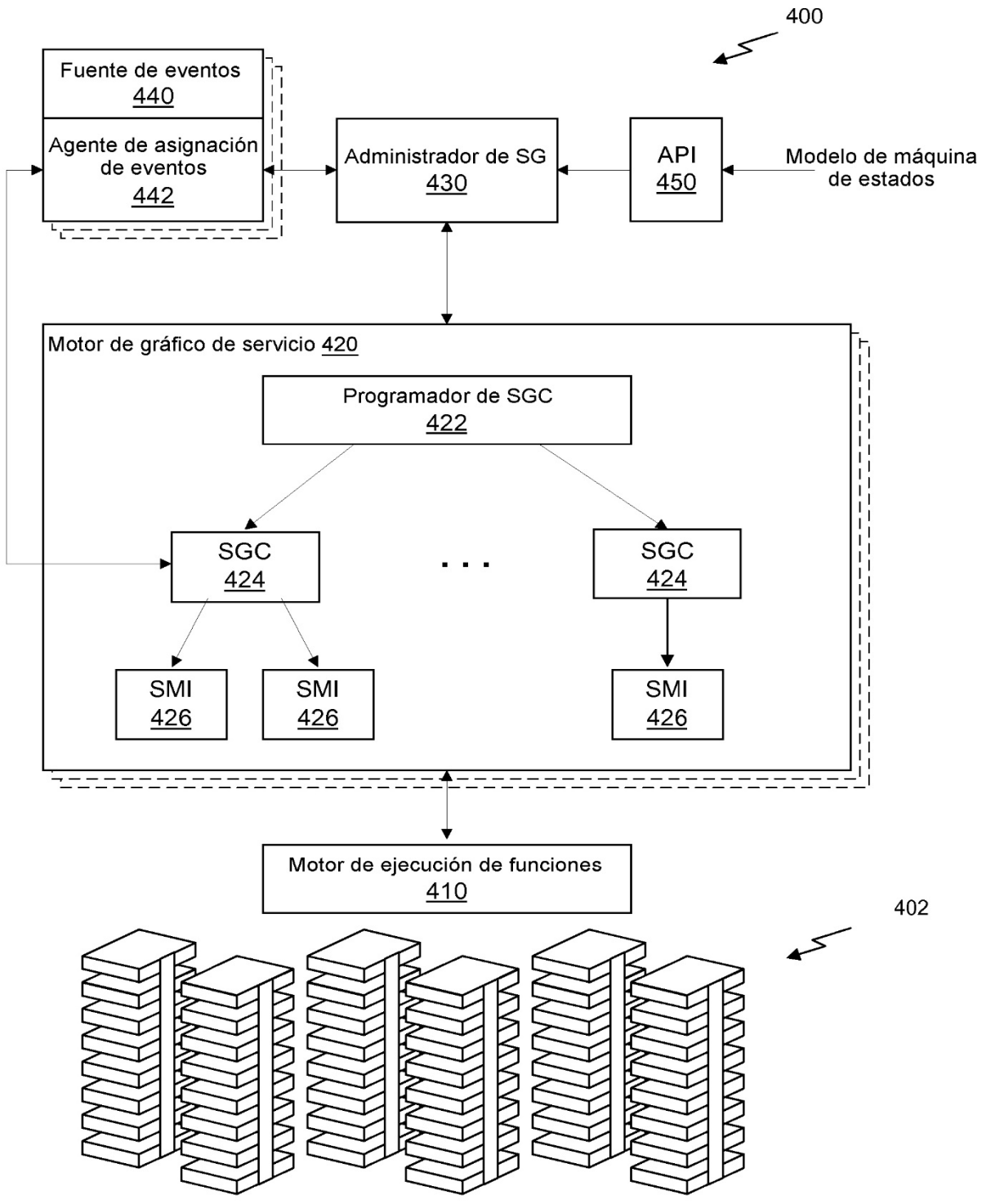
400
Fuente de eventos
440
Agente de asignación de eventos
442
Administrador de SG
430
API
450
Modelo de máquina de estados
Motor de gráfico de servicio 420
Programador de SGC
422
SGC
424
SGC
424
SMI
426
SMI
426
SMI
426
Motor de ejecución de funciones
410
402

Fig. 4A

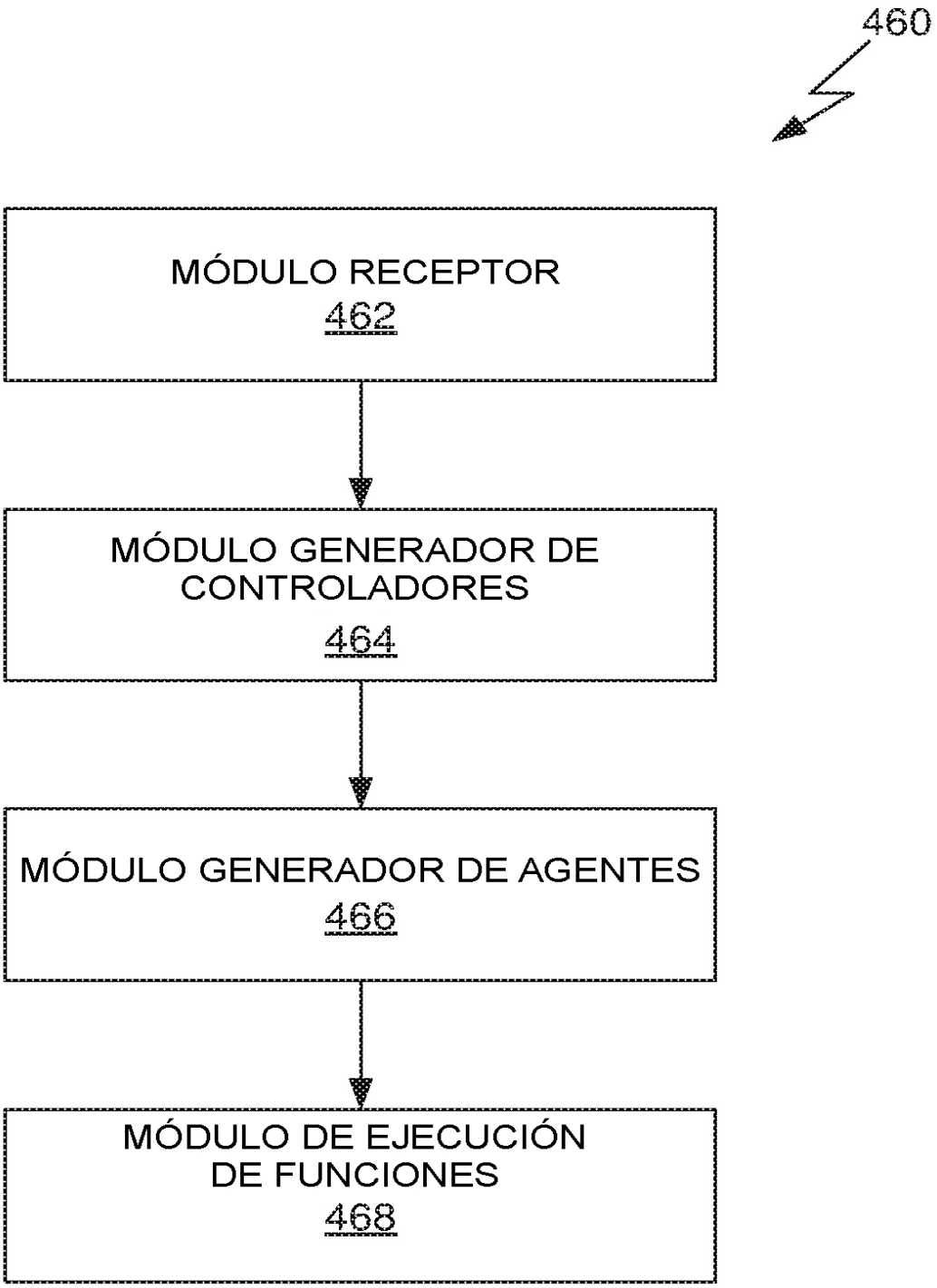
460
MÓDULO RECEPTOR
462
MÓDULO GENERADOR DE CONTROLADORES
464
MÓDULO GENERADOR DE AGENTES
466
MÓDULO DE EJECUCIÓN DE FUNCIONES
468

Fig. 4B

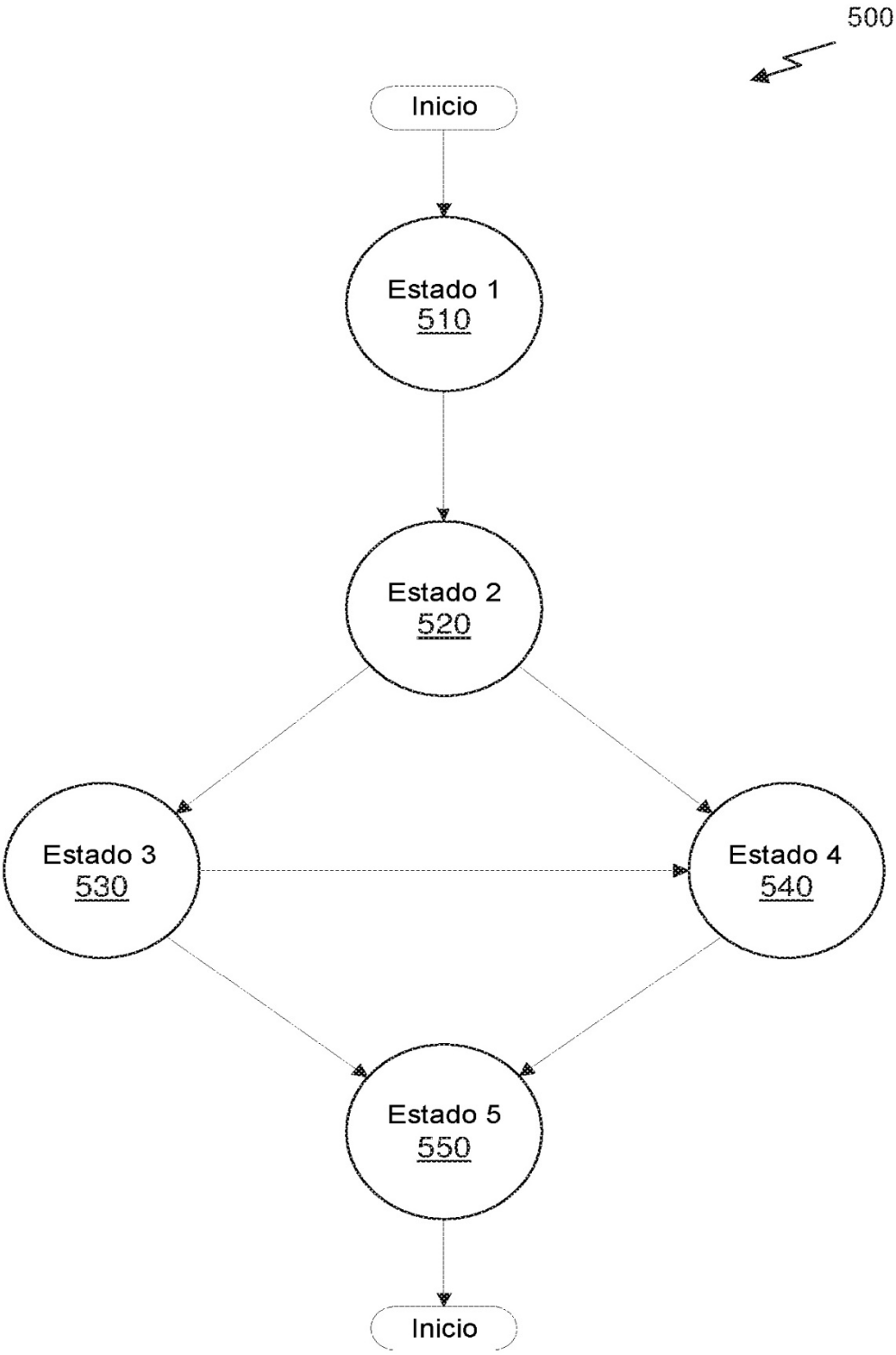
500
Inicio
Estado 1
510
Estado 2
520
Estado 3
530
Estado 4
540
Estado 5
550
Inicio

Fig. 5

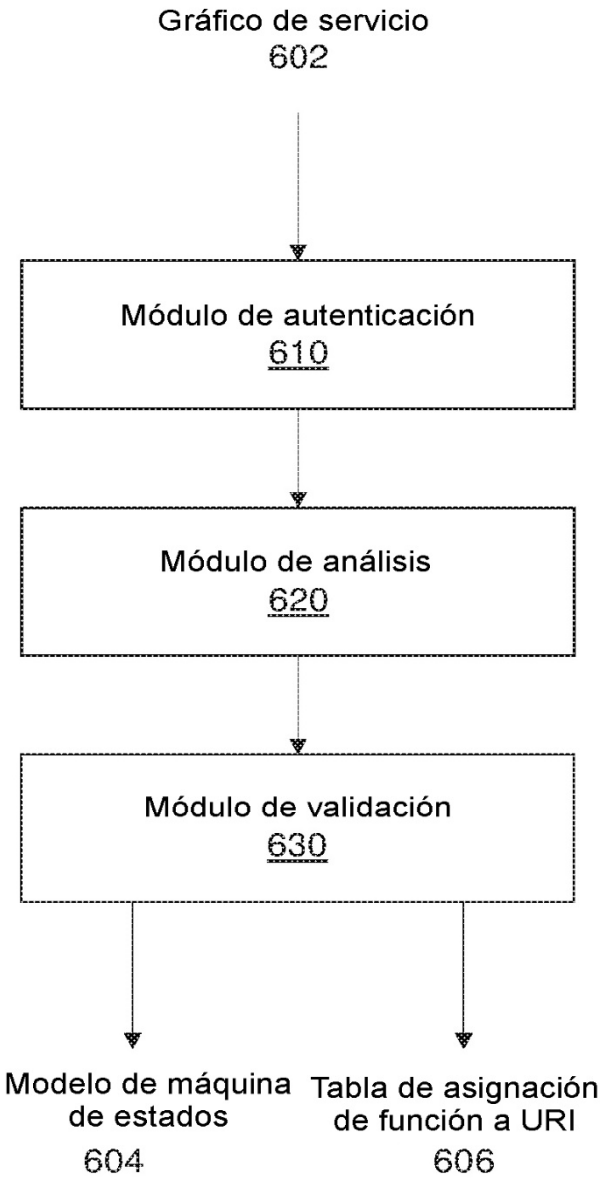
Gráfico de servicio
602
Módulo de autenticación
610
Módulo de análisis
620
Módulo de validación
630
Modelo de máquina de estados
604
Tabla de asignación de función a URI
606

Fig. 6

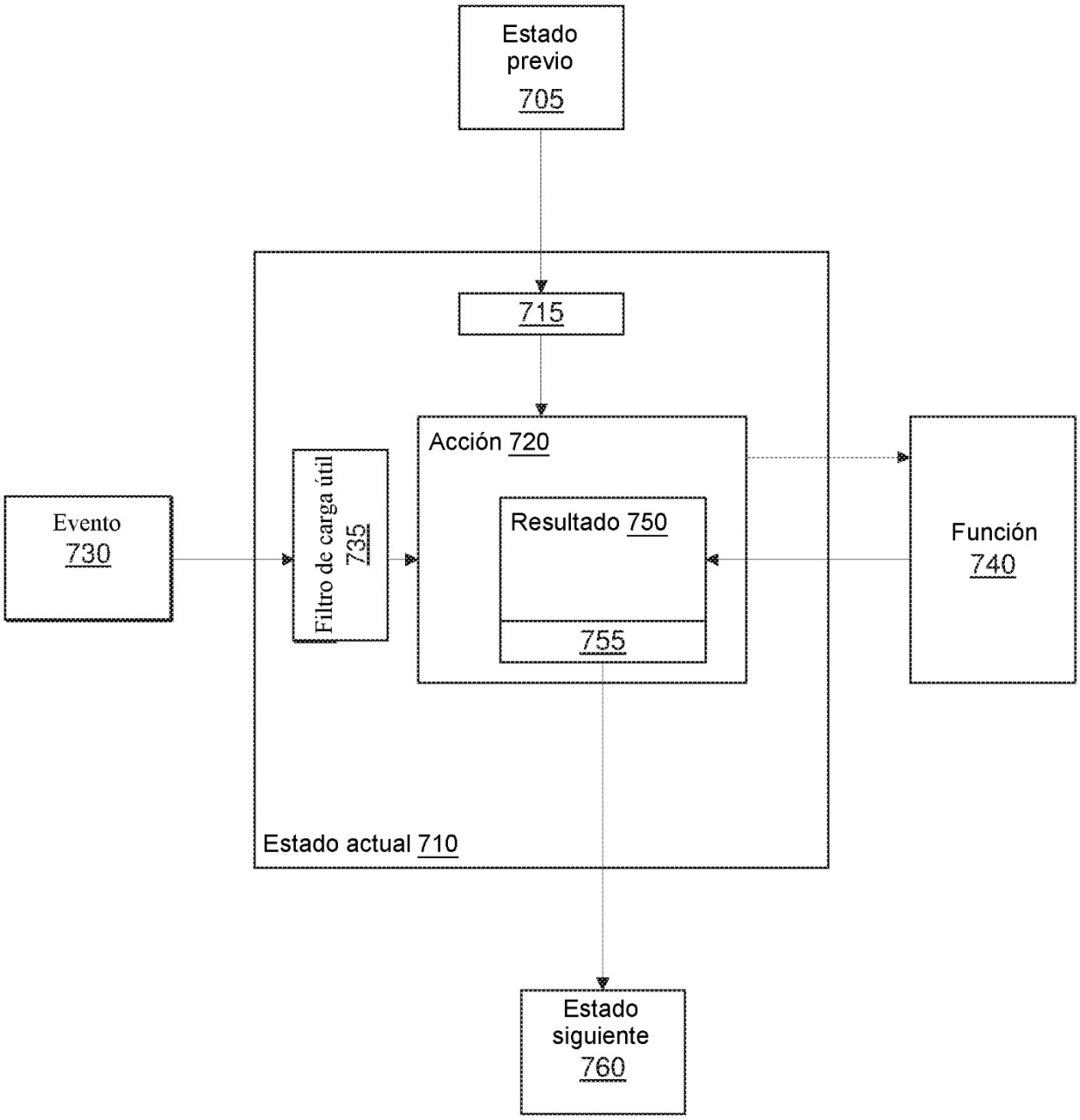
Estado previo 705
715
Acción 720
Resultado 750
755
Evento 730
Filtro de carga útil 735
Función 740
Estado actual 710
Estado siguiente 760

Fig. 7

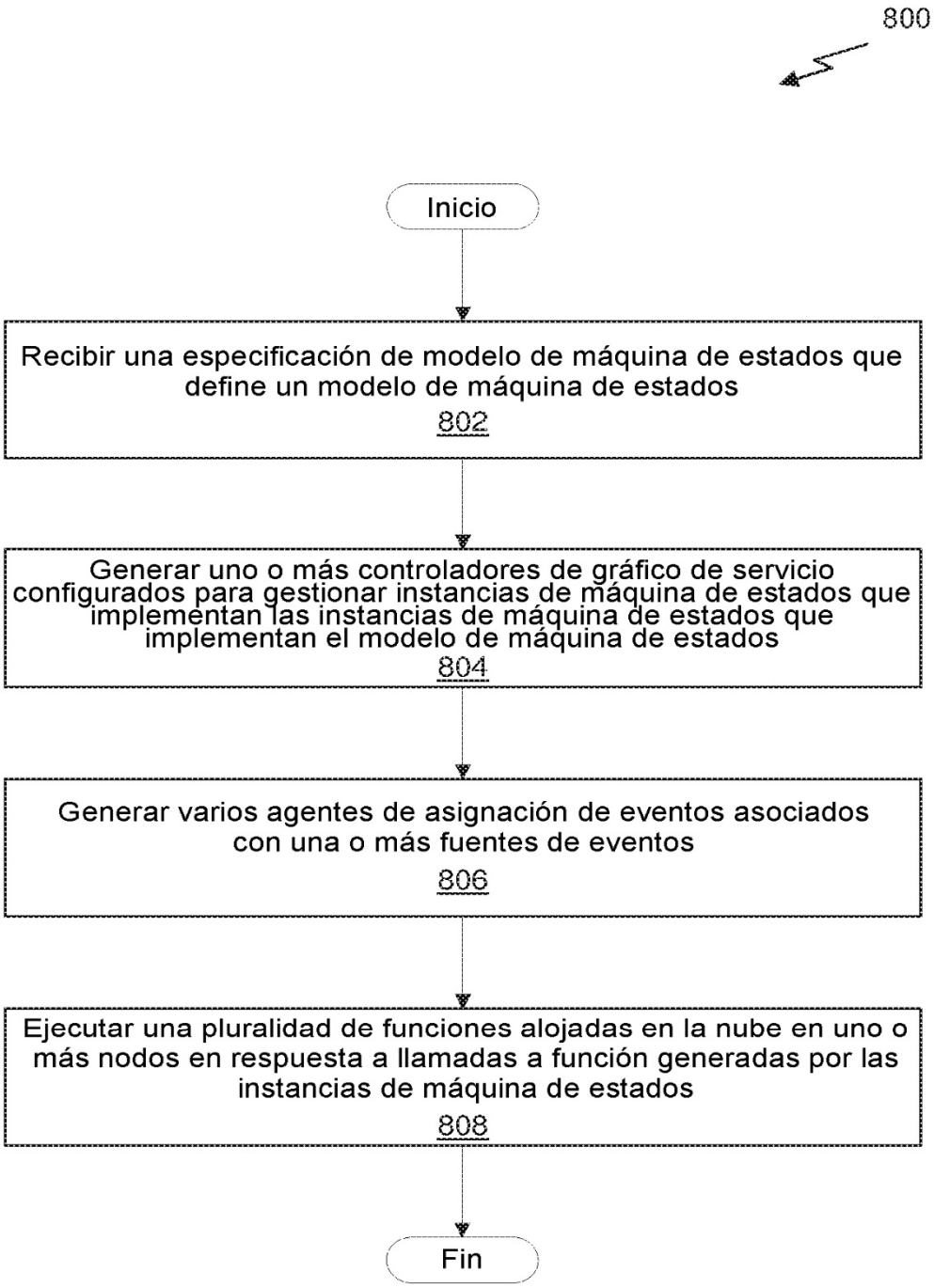
800
Inicio
Recibir una especificación de modelo de máquina de estados que define un modelo de máquina de estados
802
Generar uno o más controladores de gráfico de servicio configurados para gestionar instancias de máquina de estados que implementan las instancias de máquina de estados que implementan el modelo de máquina de estados
804
Generar varios agentes de asignación de eventos asociados con una o más fuentes de eventos
806
Ejecutar una pluralidad de funciones alojadas en la nube en uno o más nodos en respuesta a llamadas a función generadas por las instancias de máquina de estados
808
Fin

Fig. 8

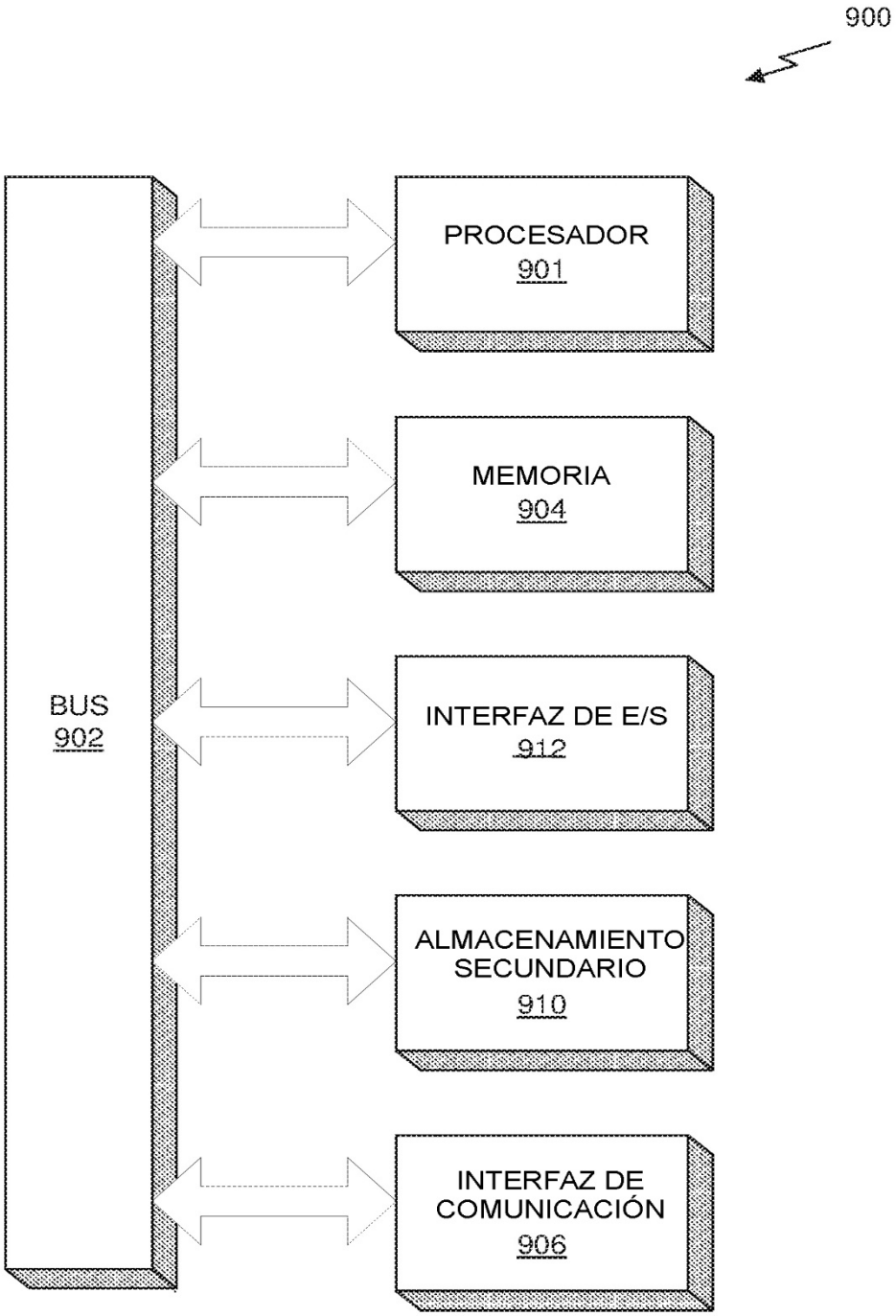
900
BUS
902
PROCESADOR
901
MEMORIA
904
INTERFAZ DE E/S
912
ALMACENAMIENTO SECUNDARIO
910
INTERFAZ DE COMUNICACIÓN
906

Fig. 9