(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 700 535**

(21) Número de solicitud: 201731034

(51) Int. CI.:

H04L 12/66 (2006.01)

(12) SOLICITUD DE PATENTE A1

(22) Fecha de presentación:

16.08.2017

(43) Fecha de publicación de la solicitud:

18.02.2019

(71) Solicitantes:

KUSANAGI, S.L. (100.0%)
C/ Carlos Arniches, 10, Tercero, Ext., Dcha.
28005 Madrid ES

(72) Inventor/es:

WATTS, James Lawrence

(74) Agente/Representante:

PONS ARIÑO, Ángel

(54) Título: **RED DISTRIBUIDA Y MÉTODO DE GESTIÓN DE LA MISMA**

(57) Resumen:

Red distribuida y método de gestión de la misma.

En este documento se detallan dos aspectos de un mismo objeto inventivo, en un primer aspecto se tiene una red distribuida que comprende en su implementación un componente especifico que es componente de registro (denominado "Discovery") que es responsable de mantener un registro de todos los componentes que figuran en la aplicación, y que forman parte del modelo de comunicación que propone el diseño de la invención. El método utilizado para guardar ese registro depende de la implementación técnica del diseño de la invención, por ello en un segundo aspecto de la invención se tiene un método de gestión de red distribuida que implementa dicho componente de registro, método que detalla el uso de dos canales de comunicación para respectivos tipos de mensajes que se transmiten entre los componentes de la red.

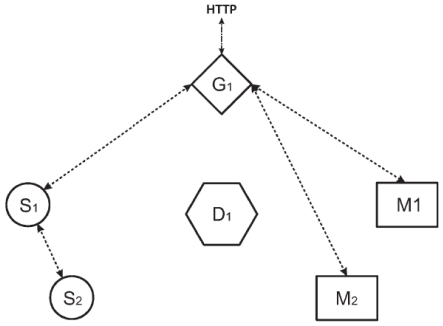

FIG. 1

ES 2 700 535 A1

RED DISTRIBUIDA Y MÉTODO DE GESTIÓN DE LA MISMA

D E S C R I P C I Ó N

OBJETO DE LA INVENCIÓN

El objeto de la invención se enmarca en el campo técnico de las tecnologías de la información y las comunicaciones.

Más concretamente el objeto de la invención va dirigido a redes de comunicaciones de las denominadas redes distribuidas.

ANTECEDENTES DE LA INVENCIÓN

Una red distribuida es una topología de red de comunicaciones en la que los nodos se vinculan unos a otros de modo que ninguno de ellos, ni siquiera un grupo estable de ellos, tiene poder de filtro sobre la información que se transmite en la red. Desaparece por tanto la divisoria entre centro y periferia característica de las redes centralizadas y descentralizadas, se tiene entonces que todos los nodos se conectan entre sí sin que tengan que pasar necesariamente por uno o varios centros.

En las redes distribuidas desaparece la división centro/periferia y por tanto el poder de filtro sobre la información que fluye por ella, asimismo una red distribuida es robusta ante la caída de nodos: ningún nodo al ser extraído genera la desconexión de otro.

El funcionamiento de una red se puede y debe monitorizar, dicha monitorización se puede realizar de forma remota de manera que se llevan a cabo una serie de chequeos a través de la red para obtener información; normalmente se utilizan protocolos de red estándar como ICMP, SNMP, TCP/UDP, HTTP, etc. Generalmente, los chequeos se ejecutan desde un servidor de monitorización centralizado y reportan información de forma inmediata, las comunicaciones normalmente son iniciadas por el propio servidor de monitorización. Mediante dichos chequeos se puede realizar una serie de comprobaciones relacionadas por ejemplo con comprobar si un equipo se encuentra levantado (“Host Alive” en inglés), latencia de las comunicaciones con un determinado equipo (“Host Latency” en inglés), comprobar si un servicio se encuentra levantado monitorizando el estado de un puerto concreto (por

ejemplo, mediante servicio HTTP usando el puerto 80), tráfico de red (preferiblemente mediante SNMP).

Otra forma de monitorizar es aquella que hace uso de agentes, es decir mediante un software ligero que permanece en ejecución en los sistemas operativos, recogiendo permanentemente datos del sistema operativo en el que se encuentra instalado. Normalmente puede acceder a niveles de información más profundos que mediante chequeos de red, además de poder monitorizar “desde dentro” las aplicaciones que se encuentran en el servidor; en este tipo de monitorización las comunicaciones suelen ser iniciadas por los propios agentes, pero también puede ser iniciada por el servidor. Mediante la monitorización con agente se suele obtener información al respecto de: uso de CPU, uso de memoria, capacidad de sistema de almacenamiento, número de procesos en ejecución, servicios levantados así como datos de monitorización interna de aplicaciones.

En este sentido se tiene el documento US9112894B2, el cual describe una plataforma de red distribuida en tiempo real así como la supervisión de seguridad para monitorizar un vínculo entre una sub-red distribuida e Internet, ésta presenta un servidor central de coordinación y de supervisión localmente accesible por diversos elementos de supervisión distribuidos. La plataforma descrita permite integrar cualquier nuevo servicio conectable o existente en la plataforma sin necesidad de reconstrucción a través de los puntos de la extensión, asegurando así la flexibilidad de uso y permitiendo la adición de nuevas funciones o servicios. La plataforma permite que el *buffering* dirija tráfico de red de alta velocidad sin pérdida de paquetes y reduciendo los recursos necesarios de procesado.

En US20140337453A1 se detalla un método que implica determinar si una petición (SNMP) del *Simple Network Management Protocol* requiere la implicación de un dispositivo dado para satisfacer la petición SNMP o de si la petición SNMP no requiere la implicación del dispositivo para satisfacer la petición SNMP por parte de un dispositivo principal para determinar que la petición del SNMP pertenece al dispositivo del usuario. Una petición no-SNMP correspondiente a la petición del SNMP se genera y se transmite del dispositivo principal al dispositivo del usuario para determinar que la petición del SNMP requiere la implicación del dispositivo del usuario para satisfacer la petición del SNMP.

En US7415038B2 se describe un método de gestión del uso de ancho de banda para un sistema de proceso de datos en redes distribuidas, el método descrito implica predecir el uso de ancho de banda en respuesta a una petición dentro del sistema de proceso de datos.

Se detalla como un *framework* de gestión de red es capaz de monitorear múltiples fuentes de paquetes de red en varias subredes dentro del sistema de procesamiento de datos distribuidos; así como el uso de *snoopers* distribuidos desde del gestor de usos de paquetes que se despliegan para supervisar las múltiples fuentes de paquetes de la red. El administrador del sistema puede solicitar el filtrado de paquetes basado en usuarios activos seleccionados o en aplicaciones activas. Una base de datos de historial de ancho de banda se genera a partir de datos de uso de ancho de banda asociados con múltiples entidades dentro del sistema de procesamiento de datos, incluyendo usuarios, aplicaciones y / o puntos finales dentro del sistema de procesamiento de datos. En respuesta a una acción solicitada dentro del sistema de procesamiento de datos, el uso de ancho de banda para la acción solicitada se puede predecir con referencia a la base de datos de ancho de banda del historial. El uso real y previsto del ancho de banda de las acciones solicitadas se puede mostrar al administrador del sistema en tiempo real.

Por lo tanto, se haría necesario poder llevar a cabo una fragmentación en conjuntos independientes y distribuir estos a través de múltiples servidores, además de mantener una comunicación eficiente entre ellos. Además, al distribuir una aplicación a través de varios servidores, se introducen retos que no existen con una aplicación que resida íntegramente en un solo servidor, como pueden ser los fallos en la red de comunicación, control de errores, pérdidas de paquetes o corrupción de contenido en la transmisión de datos.

DESCRIPCIÓN DE LA INVENCIÓN

Dada la naturaleza la invención se ha intentado mantener, en la medida de lo posible, el lenguaje utilizado por el experto medio en la materia tratada; de tal manera que a lo largo del presente documento se pueden encontrar diversos anglicismos de uso común en el campo técnico relacionado. Asimismo, cuando ha sido posible, se ha utilizado el correspondiente término en castellano, se ha defino el término en cuestión o se ha indicado la correspondencia entre términos.

A continuación se lista una serie de términos que se utilizan a lo largo de este documento, junto con una breve descripción de los mismos:

- Sistema: Se refiere a una implementación concreta de este diseño.
- “Component”: Un aspecto individual, encapsulado y aislado del sistema.
- “Host”: Una instancia de un componente del sistema.

- Cliente: Un programa externo que desea interactuar con el sistema.
- “Gateway”: El componente que actúa como punto de acceso al sistema, expuesto a una red de comunicación usando un protocolo específico.
- “Middleware”: El componente que es responsable de procesar las peticiones al sistema y formatear las respuestas para el cliente.
- “Service”: El componente que se usa para representar un dominio concreto del sistema, que procesa la lógica para peticiones dentro de ese dominio.
- “Discovery”: El componente que actúa como un registro de los demás componentes activos, su configuración, y los “Hosts” donde se encuentran localizados.
- Mensaje: Un paquete de información enviado entre componentes del sistema.
- Carga Útil de Asignación (“payload” en inglés): Un objeto, enviado por un componente del sistema, transmitido a través de uno o varios mensajes.
- Transporte: Un objeto que puede estar presente en una carga útil, que es creado por un “Gateway”, enviados a un “Service”, y posiblemente entre otros “Services”, y procesado por uno o varios “Middleware”.

El objeto de la invención se basa en mantener “quorum” entre los diferentes componentes que forman la arquitectura de la aplicación.

Para ello se propone en el siguiente documento una red distribuida así como un método de gestión de la misma. Dicha red distribuida comprende una serie de servidores “host” donde en al menos uno de los cuales se encuentran definidos una serie de componentes: Un componente de servicio (denominado “Service”) encargado de encapsular la lógica de la aplicación, un componente intermedio (denominado “Middleware”) un punto de acceso (denominado “Gateway”) y un componente de registro (denominado “Discovery”).

El objeto de la invención se centra en un componente en concreto, el denominado “Discovery”, el cual tiene la responsabilidad de recoger la configuración y el estado de los demás componentes registrados con él, y publicar a los mismos un registro de todos estos para que puedan localizarse entre ellos para comunicar. A diferencia del arte previo, donde se mantiene un punto céntrico de registro (denominado “Service Locator”), pero que la carga de consultar ese registro se distribuye a los componentes, en vez de generar un punto de congestión en ese registro central.

Todos los mensajes se manejan a través de flujos binarios sobre aquella solución disponible en el lenguaje de implementación, como puede ser “ZeroMQ” o “Nanomsg”, y utilizando

protocolos de comunicación, como puede ser IPC o TCP, en función de si los componentes se encuentran en la misma máquina física o no, para la comunicación.

Estos mensajes llevan una carga útil de asignación (“payload” en inglés), que definen el tipo del mensaje que es, y por lo tanto su objetivo.

El formato del objeto de carga más común es un "command". Estos mensajes tienen el objetivo de realizar una tarea o solicitar algo de otro componente. Las propiedades contenidas en este objeto preferiblemente deben ser al menos las siguientes:

- Nombre: El nombre del comando a llamar (cadena).
- Argumentos: Los argumentos clave / valor para el comando, si no existen argumentos esta propiedad no debe ser definida (objeto).

El formato de una respuesta a un "command" se estandariza como un "command-reply". El objetivo de estos mensajes es exclusivamente responder a un mensaje “command”. Las propiedades contenidas en este objeto deben ser al menos las siguientes:

- Nombre: El nombre del comando que procesa la respuesta (cadena).
- Resultado: Los datos proporcionados por el componente para la respuesta (objeto).

En el caso de que ocurriese un error, se podría devolver un mensaje del tipo “error” como la respuesta a un “command”. Las propiedades contenidas en este objeto deben comprender al menos la propiedad: “Código”, siendo ésta un número que represente el error ocurrido.

El transporte (“Transport”) es el objeto que debe ser creado por un componente “Gateway” en cada solicitud, y debe propagarse a través de los “Services” durante una petición. Su objetivo es de recopilar los datos proporcionados por cada “Service”, para luego ser formateado por un “Middleware” como la respuesta al “Cliente” realizando la petición. Las propiedades contenidas en este objeto pueden variar según la implementación, pero deben incluir por lo menos: “id” siendo ésta un identificador único de la petición (cadena).

Una de las características de la invención, tal y como se ha indicado anteriormente, es que un componente en concreto tiene la responsabilidad de recoger la configuración y el estado de los demás componentes registrados con él, y publicar a los mismos un registro de todos

estos para que puedan localizarse entre ellos para comunicarse. Para ello se hace uso de dos tipos de mensajes, “Heartbeat” y “Mapping”.

Los mensajes denominados “Heartbeat” (latido) se definen como mensajes que son enviados por los “Middleware” y “Service” registrados en un “Discovery”, dicho envío se lleva a cabo de manera continuada en un determinado intervalo de tiempo, y están referidos al estado del servidor (“host”) donde residen. Esto puede ser cualquier dato que el “Discovery” considere oportuno para determinar, por ejemplo, la carga al que esté expuesto ese servidor; como puede ser el uso de recursos del sistema, como son el CPU, memoria física y/o virtual, y disco.

Los mensajes de mapeo denominados “Mapping” son enviados continuadamente en un intervalo por los “Discovery” a cada uno de los componentes que estén registrados en el “Discovery; estando el mensaje relacionado con un registro de los “Middleware” y “Service” que se registraron en dicho “Discovery” y de los cuales recibe “Heartbeat”. Este registro incluye datos suficientes para que un componente pueda localizar a otro, como puede ser la dirección IP y número de puerto, y/o el protocolo de comunicación a utilizar.

DESCRIPCIÓN DE LOS DIBUJOS

Para complementar la descripción que se está realizando y con objeto de ayudar a una mejor comprensión de las características de la invención, de acuerdo con un ejemplo preferente de realización práctica de la misma, se acompaña como parte integrante de dicha descripción, un juego de dibujos en donde con carácter ilustrativo y no limitativo, se ha representado lo siguiente:

Figura 1.- Muestra un diagrama del método objeto de la invención.

Figura 2.- Muestra un diagrama del objeto de la invención donde se representa la comunicación de “Heartbeat” y “Mapping” entre el “Discovery” y los demás componentes.

REALIZACIÓN PREFERENTE DE LA INVENCIÓN

En una realización preferida de un primer aspecto de la invención mostrada en la figura 1, se tiene una red distribuida que presenta un diseño arquitectural para una aplicación de software, concretamente aquella destinada a usarse por una red de comunicación, y sobre

todo por Internet, que requiere ser fragmentada y preferentemente distribuida a través de varios servidores deferentes.

Su diseño se basa principalmente en cuatro componentes, que representan diferentes elementos arquitecturales o entidades de una aplicación. Se puede definir cualquier cantidad de componentes, siendo uno el requerimiento mínimo, y en cualquier orden o distribución. Estos se pueden repartir a través de los servidores que se deseen, siendo uno o más componentes los que residan en un servidor, y en cualquier lugar geográfico del planeta que tenga acceso a una red de comunicación común, como puede ser Internet.

Cada uno de estos componentes tiene una función y responsabilidad diferente, estando dichos componentes definidos de la siguiente manera:

1. Un componente de registro (denominado “Discovery”): Este componente es responsable de mantener un registro de todos los componentes que figuran en la aplicación, y que forman parte del modelo de comunicación que propone el diseño de la invención. El método utilizado para guardar ese registro puede depender de la implementación técnica del diseño de la invención.

2. Un punto de acceso (denominado “Gateway”): Este componente representa el acceso a la aplicación desde un programa cliente, o directamente a través de la red en que se encuentra expuesta la aplicación. Su función principal es recibir peticiones y servir respuestas según el protocolo de comunicación que esté interpretando, como por ejemplo, HTTP. No interpreta el contenido de las peticiones, sino simplemente se encarga de procesar el mensaje recibido, usando las reglas definidas por el protocolo implementado, transformando este a un objeto interno, que luego puede ser comunicado con otros componentes. Los protocolos disponibles a través de un “Gateway” dependen de la implementación técnica del diseño de la invención. Este componente debe registrarse con un “Discovery” de la aplicación para poder formar parte de la comunicación.

3. Un componente intermedio (denominado “Middleware”): Este componente al ser de tipo middleware implementa una lógica de intercambio de información entre aplicaciones de tal manera que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos, este componente intermedio se conecta con uno o varios “Gateway”, y es responsable de interpretar el contenido de las peticiones, y formular las respuestas para el “Gateway” que les proporcionó

el objeto de la petición anteriormente mencionado. Al interpretar una petición, se debe poder extraer suficiente información como para determinar a qué servicio se quiere llamar desde el “Gateway”. Cómo se determina lo que se considera la información necesaria depende de las necesidades que requiera cada implementación técnica del objeto de la invención. Tanto las peticiones como las respuestas están sujetas al protocolo que esté interpretando el “Gateway” con el que estén conectados y comunicando. Este componente debe registrarse con un “Discovery” de la aplicación para poder formar parte de la comunicación.

4. Un componente de servicio (denominado “Service”): Este componente es el encargado de encapsular la lógica de la aplicación. Cada componente de servicio o “Service” representa un dominio de la aplicación, como por ejemplo podrían ser usuarios, productos o comentarios. El objetivo es que, al fragmentar la aplicación en conjuntos independientes, estos conjuntos representen algún dominio de la aplicación, como podría ser un aspecto o proceso concreto de la misma. Cada conjunto, a su vez, puede permitir que se realicen una serie de operaciones sobre ese dominio, como podría ser en el ejemplo de usuarios, el crear un nuevo usuario, modificar un usuario existente, o eliminar un usuario. Además, un componente de servicio o “Service” puede comunicarse con otro componente de servicio o “Service” registrado en un “Discovery” común. Este componente debe registrarse con un “Discovery” de la aplicación para poder formar parte de la comunicación.

El modelo de comunicación de la invención requiere que se establezcan preferiblemente al menos dos canales de intercambio de mensajes entre componentes; dichos mensajes pueden ser de dos tipos, tal y como se aprecia en la figura 2:

1. Unos primeros mensajes (denominados latidos o “Heartbeat”): Los “Middleware” y “Service” que estén registrados con un “Discovery” envían, continuadamente en un intervalo, datos referidos al estado del servidor donde residen. Esto puede ser cualquier dato que el “Discovery” considere oportuno para determinar la carga al que esté expuesto ese servidor, como puede ser el uso de recursos del sistema, como son el CPU, memoria física y/o virtual, y disco.

2. Unos segundos mensajes (denominados de mapeo o “Mapping”): Los “Discovery” envían, continuadamente en un intervalo, un registro de los “Middleware” y “Service” que se registraron, y de los cuales recibe “Heartbeat”, a cada uno de los componentes que estén registrados al “Discovery”. Este registro incluye los datos suficientes para que un

componente pueda localizar a otro, como puede ser la dirección IP y número de puerto, y/o el protocolo de comunicación a utilizar.

La estructura de datos (denominado objeto) que se transmite entre los componentes depende de la implementación técnica del diseño de la invención.

El protocolo y/o solución técnica para el formato y/o método de transmisión de datos no es específico; sin embargo, debe permitir una comunicación asíncrona entre los componentes, de tal manera que un mensaje entre dos componentes no bloquee ese canal de comunicación a otros mensajes.

De esta manera se tiene que en una posible realización del segundo aspecto objeto de la invención, y en referencia a la figura 1, tenemos que un “Gateway" (G1) recibe una petición de una red de comunicación externa, como puede ser Internet, y lo interpreta y crea un objeto interno que representa la semántica de esa petición. El “Gateway" (G1) comunica con un primer “Middleware” (M1), pasándole el objeto para determinar el componente “Service” al que se desea llamar; entonces un primer “Middleware” (M1) analiza la petición y extrae los datos que permiten identificar al "Service", en una posible realización del objeto de la invención se puede hacer uso del protocolo HTTP.

Estos datos pueden estar en la URL, los parámetros de la cadena de búsqueda o en los encabezados (“headers”) e incluye dichos datos en un objeto que devuelve al “Gateway" (G1). Con los datos proporcionados por el primer “Middleware” (M1), el “Gateway" (G1) procede a comunicarse con un primer “Service" (S1), haciéndole llegar el objeto que contiene los datos de la petición. Al procesar su lógica, el primer “Service" (S1) también llama a un segundo "Service” (S2), haciéndole llegar el objeto recibido del “Gateway" (G1). El segundo “Service” (S2) al terminar de ejecutar su lógica, introduce sus datos en el objeto y lo devuelve al primer “Service” (S1). El primer “Service" (S1) también termina, e introduce sus datos en el objeto y lo devuelve al “Gateway” (G1). El “Gateway" (G1) entonces comunica con un segundo “Middleware” (M2) y le hace llegar el objeto recibido del primer “Service" (S1). Con los datos del objeto el segundo “Middleware” (M2) genera la respuesta y se lo devuelve al “Gateway” (G1). El “Gateway" (G1) usa la respuesta generada por el segundo “Middleware” (M2) para responder a la petición que recibió de la red de comunicación externa.

En la figura 1 aparece la figura del “Discovery” (D1) para un mejor entendimiento de una realización alternativa del objeto de la invención mostrada en la figura 2, en la que se hace uso de la implementación de un modelo de comunicación de mensajes “Heartbeat” y “Mapping” entre el “Discovery” (D1) y los demás componentes, realización basada en la realización preferente mostrada en la figura 1. En este caso se tienen además de los componentes anteriormente citados un tercer componente “Service" (S3) e interviene activamente el componente “Discovery” (D1) para poder generar los primeros mensajes (denominados latidos o “Heartbeats”) en los que “Middleware” (M1,M2) y “Services” (S1,S2,S3) que estén registrados con el “Discovery” (D1) envían el estado del servidor donde residen. Mientras que en los segundos mensajes (denominados de mapeo o “Mapping”): el “Discovery” (D1) envía un registro de los “Middleware” (M1, M2) y “Services” (S1, S2, S3) y de los cuales recibe “Heartbeats”, a cada uno de los componentes que estén registrados al “Discovery” (D1).

En una realización preferente, una vez que el “host” de un componente de “Middleware” o “Service” se ha registrado con una instancia del componente “Discovery”, mediante el respectivo registro mediante el cual el “Discovery” recibe una serie de datos (variables según implementación) siendo a partir de entonces cuando puede a comenzar recibir mensajes tipo “Heartbeats”, el “Discovery” manda mensajes tipo “Mapping” a los componentes registrados en el “Discovery” y de esta manera actualiza periódicamente su estado publicando un evento, en un intervalo preferiblemente definido en milisegundos, y dependiendo de la demanda de uso; generando de esta manera un quorum.

Este evento de envío de mensajes tipo “Mapping” y “Heartbeat” se realiza usando preferiblemente y de manera no limitativa “MessagePack” cualquier otro protocolo de transporte.

De manera preferida los mensajes tipo “Heartbeat” comprenden como la carga útil (“payload”), al menos una propiedad que permita al “Discovery” determinar si hay redundancia qué “host” elegir para un componente concreto; dicha propiedad se refiere a datos que pueden ser referidos a: CPU, memoria, red, discos, y memoria de intercambio.

Estos datos deben ser preferentemente almacenados por el componente “Discovery”, y utilizados como una referencia para determinar el estado de salud de cada “host”. Con estos datos el “Discovery” puede generar una serie de índices o “weight” generados a partir de la información comprendida en los mensajes tipo “Hearbeat” relacionados con la redundancia

de cada “host” almacenando los mismos, de tal manera que tiene a su disposición un listado de la idoneidad de cada “host” para cada componente.

REIVINDICACIONES

1. Red distribuida que comprende una pluralidad de servidores, caracterizada por que en al menos uno de los servidores reside, al menos uno de:
    - un componente de registro que a su vez comprende un registro de todos los componentes que figuran en una aplicación,
    - un punto de acceso, destinado a recibir peticiones y servir respuestas según un protocolo de comunicación que esté interpretando,
    - un componente intermedio conectado a al menos uno de los puntos de acceso
    - un componente de servicio, y
    - dos canales de mensajes que a su vez comprenden:
        - un primer canal de mensajes que conecta componentes intermedios y componentes de servicio registrados en un componente de registro para transmitir unos primeros mensajes entre componentes intermedios y componentes de servicio registrados en un componente de registro de tal manera que se envía el estado del servidor donde residen, y
        - un segundo canal de mensajes que conecta el componente de registro con cada uno de los componentes que estén registrados en dicho componente de registro para transmitir unos segundos mensajes que se envían desde el componente de registro y que comprenden un registro de componentes intermedios y componentes de servicio registrados de los cuales recibe primeros mensajes.

2. Método de gestión de una red distribuida que comprende una pluralidad de servidores, donde al menos uno de dichos servidores comprende al menos uno de los siguientes componentes:
    - un componente de registro que a su vez comprende un registro de todos los componentes que figuran en una aplicación,
    - un punto de acceso, destinado a recibir peticiones y servir respuestas según un protocolo de comunicación que esté interpretando,
    - un componente intermedio conectado a al menos uno de los puntos de acceso
    - un componente de servicio, registrado en el componente de registro, y
    - dos canales de mensajes que a su vez comprenden:

- un primer canal de mensajes que conecta componentes intermedios y componentes de servicio registrados en un componente de registro para transmitir unos primeros mensajes entre componentes intermedios y componentes de servicio registrados en un componente de registro de tal manera que se envía el estado del servidor donde residen, y
- un segundo canal de mensajes que conecta el componente de registro con cada uno de los componentes que estén registrados en dicho componente de registro para transmitir unos segundos mensajes que se envían desde el componente de registro y que comprenden un registro de componentes intermedios y componentes de servicio registrados de los cuales recibe primeros mensajes;

donde el método comprende generar una serie de índices a partir de la información comprendida en los mensajes tipo “Hearbeat”, de tal manera que tiene a disposición un listado de la idoneidad de ubicación para cada componente.

3. Método según reivindicación 2 caracterizado por que la comunicación entre los componentes es asíncrona.

4. Método según reivindicación 2 caracterizado por que cuando el servidor en el que reside un componente intermedio o componente de servicio se ha registrado con una instancia del componente de registro, se procede a actualizar periódicamente su estado publicando un evento.

5. Método según reivindicación 4 caracterizado por que el evento es enviado como un objeto serializado como carga útil (“payload”).

6. Método según reivindicación 4 caracterizado por que el evento se publica en un intervalo definido en milisegundos.

7. Método según reivindicación 4 caracterizado por que el estado comprende datos de: CPU, memoria, red, discos, y memoria de intercambio.

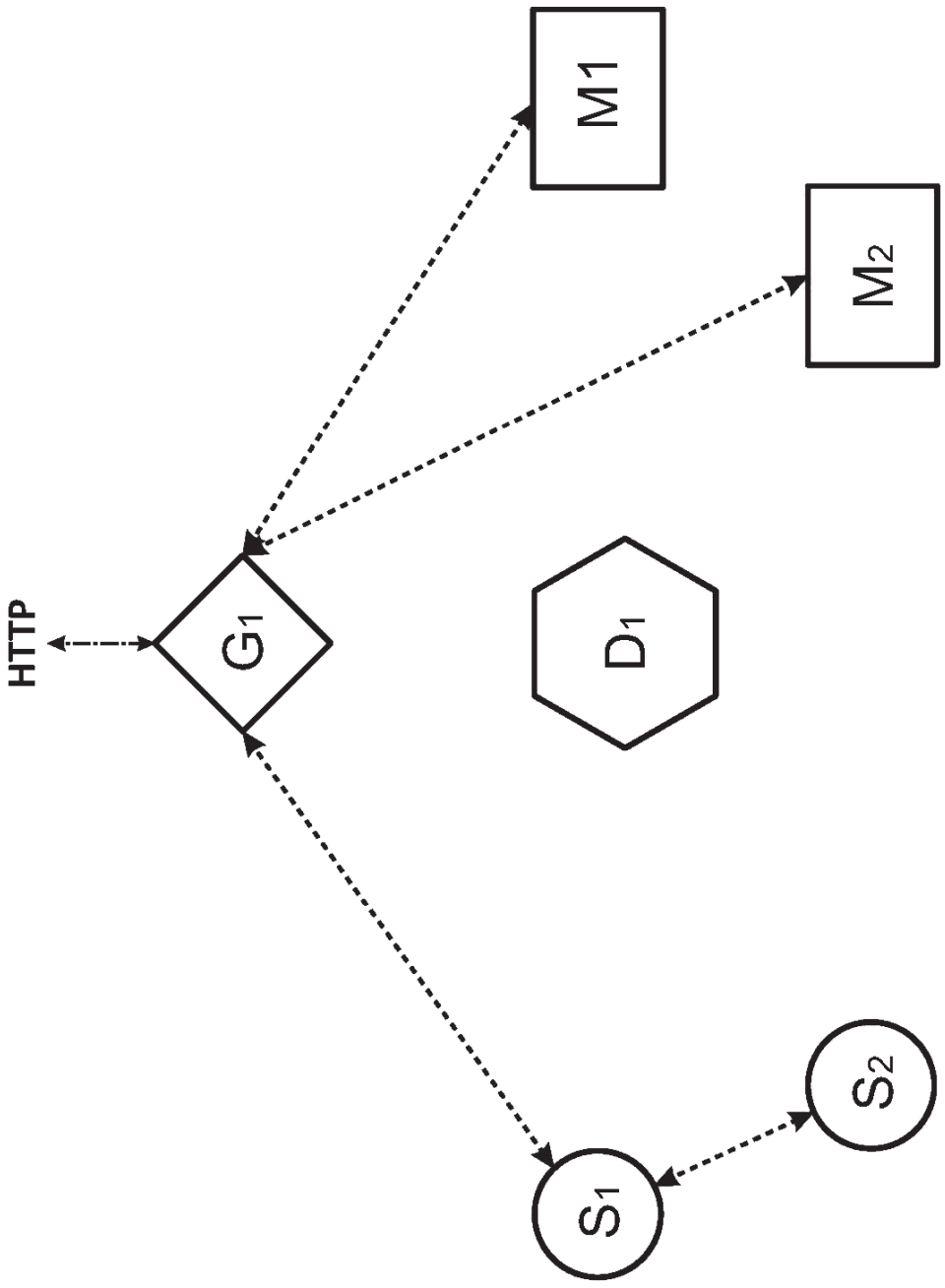

FIG. 1

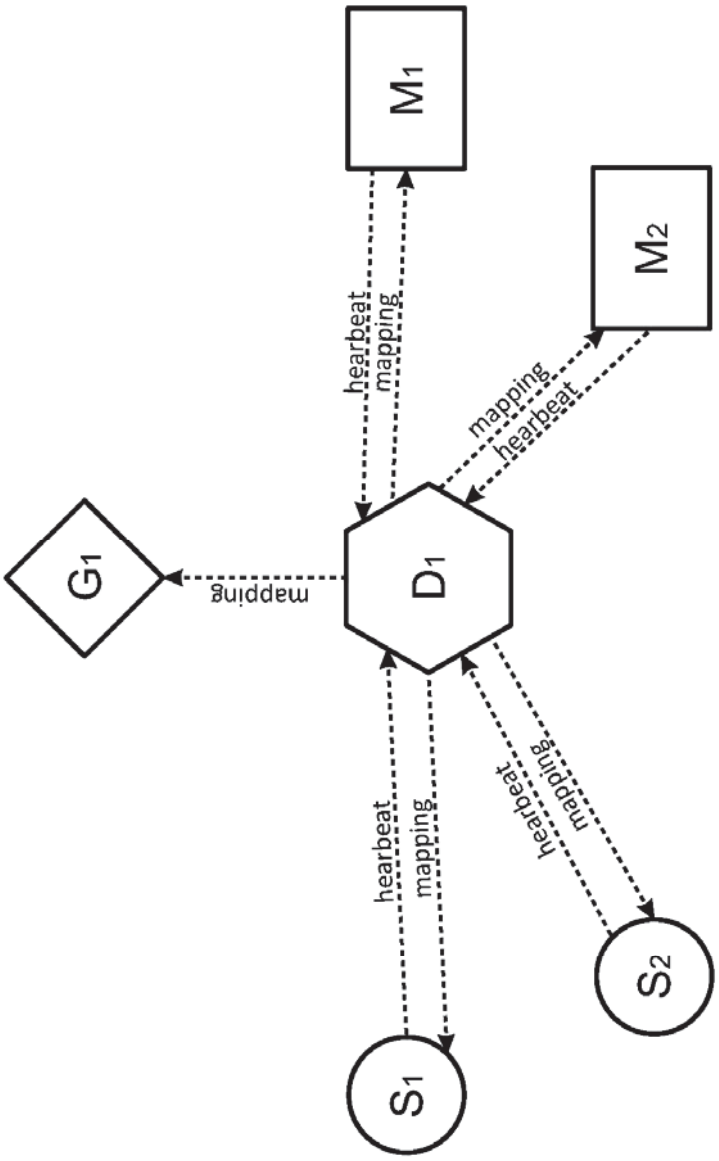
M1
M2
hearbeat
mapping
mapping
hearbeat
G1
mapping
D1
hearbeat
mapping
hearbeat
mapping
S1
S2

FIG. 2

OFICINA ESPAÑOLA
DE PATENTES Y MARCAS

ESPAÑA

(21) N.º solicitud: 201731034

(22) Fecha de presentación de la solicitud: 16.08.2017

(32) Fecha de prioridad:

INFORME SOBRE EL ESTADO DE LA TECNICA

(51) **Int. Cl. :** ***H04L12/66*** (2006.01)

DOCUMENTOS RELEVANTES

Categoría	**(56) Documentos citados**	**Reivindicaciones afectadas**
X	US 2004044727 A1 (ABDELAZIZ MOHAMED M et al.) 04/03/2004, párrafos [1 - 2]; párrafo [8]; párrafo [14]; párrafo [21]; párrafos [69 - 84]; párrafos [125 - 135]; párrafo [146]; párrafos [155 - 156]; párrafos [165 - 168]; párrafos [181 - 193]; párrafos [198 - 200]; párrafo [208]; párrafo [223]; párrafo [280]; párrafos [286 - 297]; párrafo [331]; párrafo [373]; párrafo [397]; párrafos [404 - 409]; párrafos [421 - 522]; párrafo [464]; párrafo [466]; párrafo [552]; párrafo [583]; párrafo [663]; párrafo [667]; párrafos [672 - 686]; reivindicación 1, reivindicaciones 3-7; reivindicación 12, reivindicaciones 16-17; figuras 1 - 2. figuras 4 - 5. figuras 13 - 19. figuras 29 - 37.	1
Y		2-7
Y	US 7330444 B1 (PASQUA JOSEPH) 12/02/2008, columna 1, línea 54 - columna 2, línea 16; columna 3, línea 17 - columna 4, línea 14; columna 4, línea 61 - columna 5, línea 33; columna 5, línea 49 - columna 7, línea 8; columna 6, líneas 1 - 10; figuras 1 - 7.	2-7
A	SEGGELMANN M TUEXEN. Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension; rfc6520.txt. Internet Engineering Task Force, IETF; Standard, Internet Society (ISOC), 08/02/2012 [en línea][recuperado el 01/02/2018]. Recuperado de Internet <URL: https://tools.ietf.org/html/rfc6520#page-3 >. Apartado 5.2	2-7
A	ANDERSSON L et al. LDP Specification; rfc5036.txt. Internet Engineering Task Force, IETF; Standard, Internet Society (ISOC), 01/10/2007 [en línea][recuperado el 01/02/2017]. Recuperado de Internet <URL: https://tools.ietf.org/html/rfc5036#section-2.8.2 >. Apartado 3.5.7	2
A	CARLOS A FLORES-CORTÉS et al. An Adaptive Middleware to Overcome Service Discovery Heterogeneity in Mobile Ad Hoc Environments. IEEE DISTRIBUTED SYSTEMS ONLINE, 01/07/2007, Vol. 8, Nº 7, Páginas 1-11 [en línea][recuperado el 02/02/2017]. Recuperado de Internet <URL: http://ieeexplore.ieee.org/abstract/document/4292039/ >, ISSN 1541-4922, <DOI: 10.1109/MDSO.2007.41>	1-7

Categoría de los documentos citados

X: de particular relevancia
Y: de particular relevancia combinado con otro/s de la misma categoría
A: refleja el estado de la técnica
O: referido a divulgación no escrita
P: publicado entre la fecha de prioridad y la de presentación de la solicitud
E: documento anterior, pero publicado después de la fecha de presentación de la solicitud

El presente informe ha sido realizado

☒ para todas las reivindicaciones ☐ para las reivindicaciones nº:

Fecha de realización del informe	**Examinador**	**Página**
09.02.2018	J. M. Vazquez Burgos	1/2

Documentación mínima buscada (sistema de clasificación seguido de los símbolos de clasificación)

H04L

Bases de datos electrónicas consultadas durante la búsqueda (nombre de la base de datos y, si es posible, términos de búsqueda utilizados)

INVENES, EPODOC, WPI, INTERNET