

19



OFICINA ESPAÑOLA DE  
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 604 469**

51 Int. Cl.:

**H04L 29/06** (2006.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

86 Fecha de presentación y número de la solicitud internacional: **06.05.2013 PCT/EP2013/059387**

87 Fecha y número de publicación internacional: **14.11.2013 WO13167532**

96 Fecha de presentación y número de la solicitud europea: **06.05.2013 E 13722726 (0)**

97 Fecha y número de publicación de la concesión europea: **24.08.2016 EP 2847967**

54 Título: **Procedimiento y sistema para carga anticipada de datos**

30 Prioridad:

**09.05.2012 FI 20125500**

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

**07.03.2017**

73 Titular/es:

**AALTO-KORKEAKOULUSÄÄTIÖ (100.0%)  
Otakaari 1  
02150 Espoo, FI**

72 Inventor/es:

**MANNER, JUKKA y  
WANG, LE**

74 Agente/Representante:

**LEHMANN NOVO, María Isabel**

ES 2 604 469 T3

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

## DESCRIPCIÓN

Procedimiento y sistema para carga anticipada de datos

**Sector técnico de la invención**

La invención se refiere en general al procesamiento de datos y más particularmente a la recuperación de contenido.

**5 Técnica anterior**

En lo que sigue, contenido significa cualquier contenido que se cargue en ráfagas. En particular, el contenido se puede cargar desde internet e incluir por lo menos una página web dinámica que sea una página web que está preparada con información (contenido y/o distribución) nueva para cada visualización individual. En este contexto, el término contenido se referiría específicamente a contenido web.

10 Normalmente, el contenido se carga en un terminal (el término "terminal" se refiere a continuación, en particular pero no necesariamente, a un terminal móvil o dispositivo móvil pero puede ser un dispositivo de sobremesa o portátil o cualquier otro dispositivo cliente) a un navegador, elemento a elemento. En primer lugar, se carga una descripción de la página dinámica, tal como una página de lenguaje de marcado de hipertexto (HTML, Hypertext Markup Language). El navegador analiza sintácticamente la página HTML y presenta a continuación solicitudes independientes para cada elemento que, con la descripción de la página dinámica, forman por lo tanto el contenido dinámico. Actualmente, es normal que un solo contenido contenga de 30 a 100 objetos web que tienen que ser cargados, no pocas veces desde emplazamientos diferentes. Los objetos web pueden incluir imágenes, banderolas publicitarias, animaciones flash, etc.

20 El consumo de energía de un terminal que está cargando un contenido es normalmente mayor durante la carga, que cuando el terminal está en su estado de reposo, es decir, cuando el usuario está leyendo el contenido que ya se ha cargado.

Además, puede ocurrir que la propia carga tarde más de lo previsto, en particular cuando algunos de los objetos web no se pueden recuperar. Incluso si los objetos web se pudieran recuperar, puede ocurrir que tengan que ser recuperados de un servidor remoto, lo cual tiende a aumentar la duración de la recuperación, especialmente si la recuperación se lleva a cabo sobre una conexión lenta, que puede ser una conexión inalámbrica.

30 Existen algunos procedimientos que pueden ser utilizados para reducir el consumo de energía durante la carga de contenidos. En los estándares GPRS o 3GPP, el terminal inalámbrico que carga un contenido dinámico debería entrar en estado de reposo después de que transcurra un tiempo predeterminado si el contenido solicitado no está siendo transmitido al terminal. Dado que cambiar el terminal al estado de reposo requiere cierta señalización, se considera (ver "Network Efficiency Task Force Fast Dormancy Best Practices" (mejores prácticas para inactividad rápida del grupo de trabajo de eficiencia en la red), GSM Association, mayo de 2010) que algunos modelos móviles entran agresivamente en estado de reposo para extender la vida útil de la batería y minimizar la congestión de red, pero pueden provocar problemas de carga de señalización en la red del operador. Durante la navegación, el dispositivo móvil no tiene la opción de entrar en estado de reposo o en estado de baja energía debido a las solicitudes y respuestas HTTP que se producen en ambos sentidos.

40 El modelo de transición de estado RRC se ha mantenido más bien inalterado a través de varias versiones, a saber, la versión 99, la versión 05 y la versión 06. Perala et al. presentan en "Theory and practice of RRC state transitions in UMTS networks" (teoría y práctica de transiciones de estado RRC en redes UMTS), en Proc. Fifth IEEE Broadband Wireless Access Workshop (BWA), localizado conjuntamente con IEEE GLOBECOM 2009, Hawaii, USA, noviembre a diciembre de 2009, una metodología para descubrir parámetros de configuración RRC sin involucración o cooperación del operador, lo que muestra cómo las configuraciones de red del operador pueden diferir drásticamente entre sí.

45 Feng Qian et al., en "Characterizing Radio Resource Allocation for 3G Networks" (caracterización de asignación de recursos radioeléctricos para redes 3G), en ACM Internet Measurement Conference (IMC) 2010, Melbourne, Australia, y Feng Qian et al. en "TOP: Tail Optimization Protocol for Cellular Radio Resource Allocation" (TOP: optimización de protocolo de cola para asignación de recursos radioeléctricos celulares) en IEEE ICNP 2010, Kyoto, Japón, caracterizan el impacto de las configuraciones de máquina del estado operativo y muestran que el periodo de tiempo de cola se adapta al valor del temporizador de inactividad antes de una degradación del estado. Durante un tiempo de cola, el equipo del usuario sigue ocupando canales de transmisión y su consumo de potencia radioeléctrica se mantiene al nivel correspondiente del estado, incluso aunque no haya tráfico transmitido durante el periodo.

50 Signals Research Group muestra en "Reducing the impact of smartphone-generated signaling traffic while increasing the battery life of the phone through the use of network optimization techniques" (reducción del impacto del tráfico de señalización generado por teléfonos inteligentes aumentando al mismo tiempo la vida útil de la batería del teléfono mediante la utilización de técnicas de optimización de redes) (mayo de 2010) cómo el tráfico de señalización generado por los teléfonos inteligentes afecta a la vida útil de la batería y provoca congestión en la red.

Además del documento de Fast Dormancy Best Practices (mejores prácticas para inactividad rápida), "Understanding Smartphone Behavior in the Network" (comprensión del comportamiento de los teléfonos inteligentes en la red), de los laboratorios Nokia Siemens Networks Smart (2011), explica asimismo el problema del temporizador y muestra que la inactividad rápida es una de las soluciones, que fuerza a los dispositivos a regresar rápidamente al estado de consumo de baja energía para ahorrar energía y reducir la carga de tráfico.

La memoria US 2005/0182826 A1 da a conocer un procedimiento y un aparato para comunicaciones de datos sobre una red inalámbrica. Un usuario contacta con una pasarela de internet sobre la red inalámbrica. Cuando el usuario solicita un recurso de internet, envía a la pasarela un comando HTTP mejorado. El comando HTTP mejorado hace que la pasarela de internet acceda a un archivo de recursos y obtenga no sólo el recurso solicitado sino asimismo la totalidad de los recursos incorporados en el recurso solicitado. El recurso solicitado y los recursos incorporados son empaquetados en un archivo y enviados al usuario.

### Objetivo de la invención

Los inventores han descubierto que es ventajoso que los objetos referidos que pertenecen a un contenido sean recuperados primero por un servidor y enviados a continuación al terminal que solicita el contenido, solamente después de que el servidor haya recibido la totalidad, o por lo menos la mayor parte de los objetos referidos pertenecientes al contenido. Esta disposición permite una transmisión de datos más eficiente del servidor al terminal. La solución se ha dado a conocer en mayor detalle en la solicitud de patente PCT/EP2012/053214, aún no publicada en la fecha de la primera presentación de patente de la presente invención.

La disposición descrita en el párrafo anterior puede ser utilizada para mejorar la eficiencia energética en el terminal, cuando se reciben contenidos en el terminal.

Las soluciones mencionadas en la sección anterior pueden requerir modificaciones en el terminal. Aunque la solución propuesta puede reducir significativamente el consumo de energía, sigue existiendo la necesidad de más flexibilidad de implementación del concepto de empaquetamiento. Tener reprogramado el modo de funcionamiento de un terminal -en particular, de un terminal móvil-, puede ser difícil en vista del enorme número de diferentes sistemas operativos y versiones, fabricantes y tipos de dispositivos involucrados. Además, reprogramar un terminal no se considera un procedimiento transparente dado que sería necesario un procesamiento de datos diferente en el servidor, en función de si un terminal particular ha sido o no reprogramado.

Por lo tanto, un objetivo de la presente invención es aumentar la transparencia al descargar contenido de un servidor a un terminal.

### Resumen de la invención

El objetivo se puede conseguir con un servidor acorde con la reivindicación independiente 1, con un procedimiento acorde con la reivindicación independiente paralela 5 y con un sistema acorde con la reivindicación independiente paralela 9.

Las reivindicaciones dependientes describen diversos aspectos ventajosos del servidor y del procedimiento.

### Ventajas de la invención

Con un servidor que está configurado para:

a) en respuesta a recibir de un terminal una solicitud HTTP de contenido, responder al terminal con una respuesta a HTML que contiene instrucciones configuradas para establecer un canal de comunicación bidireccional en el terminal para una comunicación entre el servidor y el terminal como un WebSocket;

b) cargar el contenido que comprende por lo menos un objeto referente y una serie de objetos referidos que pertenecen a dicho contenido, donde el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos son objetos web referidos en dicha página de lenguaje de hipertexto;

c) generar un objeto referente modificado sustituyendo en el objeto referente aquellos enlaces con objetos referidos que son enlaces externos, con enlaces modificados que apuntan a una versión del mismo objeto almacenada localmente;

d) enviar los objetos referidos al terminal sobre el canal de comunicación bidireccional; y

e) enviar al terminal el objeto referente modificado, de tal modo que los enlaces modificados en el objeto referente modificado apuntan a objetos referidos ya enviados al terminal, solamente después de que el servidor ha enviado los objetos referidos al terminal,

la descarga del contenido al terminal se puede realizar de manera completamente transparente en el terminal, lo que no era el caso con ninguna de las soluciones mencionadas anteriormente.

Análogamente, con un procedimiento para hacer funcionar un servidor, que comprende las etapas de:

a) en respuesta a recibir de un terminal una solicitud HTTP de contenido, responder al terminal con una respuesta HTML que contiene instrucciones configuradas para establecer un canal de comunicación bidireccional en el terminal para una comunicación entre el servidor y el terminal como un WebSocket;

5 b) cargar el contenido que comprende por lo menos un objeto referente y una serie de objetos referidos que pertenecen a dicho contenido, donde el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos son objetos web referidos en dicha página de lenguaje de hipertexto;

c) generar un objeto referente modificado sustituyendo en el objeto referente aquellos enlaces con objetos referidos que son enlaces externos, con enlaces modificados que apuntan a una versión del mismo objeto almacenada localmente;

10 d) enviar los objetos referidos al terminal sobre el canal de comunicación bidireccional; y

e) enviar al terminal el objeto referente modificado, de tal modo que los enlaces modificados en el objeto referente modificado apuntan a objetos referidos ya enviados al terminal, solamente después de que se haya llevado a cabo la etapa d),

15 la descarga del contenido al terminal se puede realizar de manera completamente transparente en el terminal, lo que no era el caso con ninguna de las soluciones mencionadas anteriormente.

La idea principal de los inventores es que, utilizando el servidor y el procedimiento, el terminal recibe una respuesta del servidor. Dado que la respuesta contiene instrucciones configuradas para establecer un canal de comunicación bidireccional en el terminal para la comunicación entre el servidor y el terminal, los objetos referidos se pueden transferir sobre el canal de configuración bidireccional entre el servidor y el terminal, es decir no sobre el mismo protocolo que se ha utilizado para transmitir solicitudes de contenido y respuestas a dichas solicitudes. Esta separación hace posible transmitir los objetos referidos juntos en una sola transferencia. Esto puede sorprendentemente ser utilizado dado que cuando el canal de configuración direccional no está activo, tampoco el terminal necesita estar activo, para ahorrar energía durante dichos períodos no activos. Esto es diferente a la segmentación HTTP. La segmentación HTTP permite al terminal solicitar y recibir múltiples objetos del mismo servidor de origen sobre la misma conexión de transporte, mientras que en la idea de los inventores todo el contenido se reúne desde potencialmente múltiples fuentes y a continuación se envía a través de una única conexión de transporte. Según la idea de los inventores, hay muchas menos solicitudes HTTP GET necesarias desde el terminal, dado que el servidor envía directamente todos los objetos al cliente.

20 Cuando el servidor está configurado para enviar al terminal el objeto referente modificado solamente después de que el servidor ha enviado al terminal los objetos referidos, el terminal tiene ya los objetos referidos disponibles localmente y, por lo tanto, puede utilizar sin retardo el objeto referente.

El servidor está configurado para establecer el canal de comunicación bidireccional como un WebSocket. El WebSocket está siendo estandarizado por W3C, y el protocolo de WebSocket ha sido estandarizado por el IETF como RFC 6455. "File API: Directories and System" (API de archivos: directorios y sistema) es actualmente un borrador de trabajo de W3C, que define una API para navegar por jerarquías del sistema de archivos, y define un medio mediante el cual un agente de usuario puede exponer a aplicaciones web, en aislamiento de procesos, secciones de un sistema de archivos local del usuario (ver <http://www.w3.org/TR/file-system-api/>). Ventajosamente, las instrucciones de establecimiento de WebSocket y de almacenamiento de objetos referidos se generan todas en una etapa de proceso (P201, ver la figura 4). Actualmente, WebSocket está soportado por casi todos los navegadores de sobremesa, y entre los navegadores móviles está soportado por IOS Safari y Opera Mobile (ver <http://caniuse.com/websockets>). Parece que actualmente la API de archivos está soportada solamente por Google Chrome y ningún navegador móvil la soporta (ver <http://www.html5rocks.com/en/tutorials/file/filesystem/>). En la actual implementación de los inventores, el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos son objetos web a los que se hace referencia en dicha página de lenguaje de hipertexto.

45 Si el servidor está configurado para enviar en un paquete al terminal los objetos referidos, la comunicación sobre el WebSocket se puede realizar de manera más eficiente dado que el envío del paquete se puede realizar de una vez, en contraste con enviar los objetos referidos de uno en uno. Además, el terminal tiene ahora la opción de entrar en el estado de consumo de baja energía mientras el servidor está cargando y empaquetando, y antes de que éste envíe la respuesta al terminal. Las opciones aumentan aún más cuando se utiliza este concepto junto con el concepto de inactividad rápida que se ha mencionado anteriormente. En el caso más ventajoso, el paquete se envía solamente después haberse enviado instrucciones de envío para establecer un WebSocket e instrucciones para guardar localmente en el terminal los objetos referidos.

55 Un sistema comprende por lo menos un servidor según el aspecto anterior de la presente invención, y una serie de terminales que son preferentemente terminales móviles. El sistema está configurado para utilizar dicho por lo menos un servidor para responder a las solicitudes de contenido que han sido enviadas por los terminales.

**Lista de dibujos**

A continuación, la invención se describe en mayor detalle haciendo referencia al ejemplo de los dibujos adjuntos en las figuras 1 a 5, de las cuales:

- la figura 1 muestra el servidor estableciendo un WebSocket entre el terminal y el servidor;
- 5 la figura 2 muestra el servidor cargando objetos sobre protocolo HTTP y enviando los objetos al terminal por medio del WebSocket;
- la figura 3 muestra el servidor enviando al terminal el objeto referente modificado para que el terminal cargue a continuación los objetos en el navegador;
- la figura 4 muestra la señalización entre el terminal, el servidor y un servidor remoto; y
- 10 la figura 5 muestra el principio de generación de un objeto referente modificado.

Los mismos numerales de referencia y símbolos se refieren a los mismos componentes en todas las figuras.

**Descripción detallada**

La figura 1 muestra el servidor 120 estableciendo un WebSocket WS entre el terminal 10 y el servidor 120. El servidor 120 puede ser, en particular, un servidor intermediario (intermediario de red) pero, alternativamente, puede ser un servidor de contenido. El terminal 10 puede ser cualquier dispositivo cliente, en particular puede ser un dispositivo inalámbrico o móvil o un terminal inalámbrico o móvil, tal como un teléfono inteligente o una PDA, o un dispositivo informático, tal como en particular un ordenador de sobremesa, un ordenador portátil o una tableta.

El terminal 10 y el servidor 120 tienen ambos por lo menos una pila de protocolos adecuada para comunicación bidireccional. En particular, en la pila de protocolos, se han implementado preferentemente control de acceso al medio (MAC, Media Access Control), protocolo de internet (IP, Internet Protocol), protocolo de control de transferencia (TCP, Transfer Control Protocol) y protocolo de transferencia de hipertexto (HTTP, Hypertext Transfer Protocol).

La realización preferida de la invención se explica a continuación haciendo referencia al diagrama de señalización de la figura 4 y a las ilustraciones de pila de protocolos de las figuras 1 a 3.

25 El terminal 10 envía al servidor 120 una solicitud M201 que es, en el caso más preferente, una solicitud HTTP, en particular la solicitud M201 puede ser una solicitud de una página web, pero en principio puede ser una solicitud de cualquier contenido C.

En respuesta, el servidor 120 genera (etapa P201) y envía la respuesta M203 al terminal 10. La respuesta M203 comprende en el caso más preferentemente una página en lenguaje de marcado de hipertexto (HTML) que comprende instrucciones para establecer un WebSocket WS entre el terminal 10 y el servidor 120, e instrucciones para almacenar objetos referidos B1, B2, B3, ... en el terminal 10. El terminal 10 establece el WebSocket WS en la etapa P203. Además, si el terminal 10 recibe los objetos referidos B1, B2, B3, ... en por lo menos un paquete, la respuesta M203 puede contener instrucciones para que el terminal 10 desempaque los objetos referidos B1, B2, B3, ... del paquete.

35 El servidor 120 envía un conjunto de solicitudes M205 a por lo menos un servidor web 140 (o a cualquier otro servidor remoto) y recibe un conjunto de respuestas M207. Se debe entender que puede haber uno o varios servidores remotos o servidores web 140 cada uno de los cuales recibe por lo menos una solicitud M205.

En el caso más preferente, las solicitudes M205 son las solicitudes HTTP y las respuestas M207 son respuestas HTTP en el caso más preferente. Las respuestas M207 pueden comprender por lo menos un objeto referente A y por lo menos uno, pero en el caso más preferente una serie de objetos referidos B1, B2, B3, ....

40 En el presente ejemplo (ver la figura 5), el objeto referente A es la página web original y el objeto referido B1 es el objeto www.example.com/image.png. En el ejemplo, el objeto referente A y los objetos referidos B1 componen el contenido solicitado C.

**Ejemplo:** la página web original

```

45     <!DOCTYPE html>
        <html lang="en">
            <head>
                <meta charset="utf-8">
                <title>Gatos Que Programan demo</title>
    
```

```

</head>
<body>
  <div>
    <header>
      <h1 id="logo"><a href="#">Gatos Que Programan</a></h1>
    </header>
    <!-- original line -->
    <img src='http://www.example.com/image.png' />
    <!-- original line -->
  </div>
  <div>
    <h2>Paris, France</h2>
    <p> Un gato que programa ordenadores es un animal particularmente
    interesante. Para quienes aún no lo saben, este texto ha sido escrito asimismo
    por el gato inteligente.</p>
  </div>
</div>
</body>
</html>

```

El objeto referente A se puede almacenar localmente en el servidor 120, de tal modo que no tiene que ser recuperado necesariamente, en particular cuando el servidor 120 es un servidor web. Sin embargo, si el servidor 120 es un servidor intermediario, entonces en el caso más probable el servidor 120 tiene que solicitar asimismo el objeto referente A al servidor remoto 140, que puede ser un servidor web o cualquier otro servidor.

En la etapa P205 el servidor 120 analiza sintácticamente el objeto referente A, es decir, la página HTML principal, y sustituye la totalidad o por lo menos algunos enlaces externos L1, L2, ... Con enlaces internos ML1, ML2, ..., es decir, enlaces con el prefijo "filesystem". El resultado es el objeto referente modificado AM.

En el presente ejemplo, el enlace L1 pasa a ser por lo tanto el enlace modificado ML1: "<http://www.example.com/image.png>" se cambia a "<filesystem:http://www.example.com/temporary/image.png>". Se observa que es suficiente añadir al enlace externo L1, L2, ... tan sólo un sufijo para obtener el enlace modificado ML1, ML2, ...; en particular, el sufijo puede ser tan simple como el texto "filesystem:" o cualquier otro identificador adecuado.

Ejemplo (continuación): la página web modificada

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Gatos Que Programan demo</title>
</head>
<body>
  <div>
    <header>
      <h1 id="logo"><a href="#">Gatos Que Programan</a></h1>
    </header>

```

```

<!-- modified line -->
    <img src='filesystem:http://www.example.com/temporary/image.png' />
<!-- modified line -->
    <div>
5      <h2>Paris, France</h2>
        <p> Un gato que programa ordenadores es un animal particularmente
        interesante. Para quienes aún no lo saben, este texto ha sido escrito asimismo
        por el gato inteligente.</p>
    </div>
10    </div>
    </body>
    </html>

```

En la etapa P207 el servidor 120 empaqueta parte o la totalidad de los objetos referidos B1, B2, B3, ... El empaquetamiento se puede llevar a cabo utilizando un programa de archivo que no utiliza compresión (tal como TAR) o que utiliza compresión (tal como ZIP).

A continuación, el servidor 120 envía al terminal 10 dicho por lo menos un paquete utilizando el WebSocket WS (indicado como mensaje M209). En la etapa P209, el terminal 10 recibe el paquete y guarda localmente el paquete, extrae los objetos B1, B2, B3, ... del paquete y los almacena localmente en el sistema de archivos del navegador.

Se debe entender que el empaquetamiento de la etapa P207 es opcional. Si los objetos B1, B2, B3, ... no se almacenan por lo menos en un paquete, se transmiten de uno en uno desde el servidor 120 al terminal 10 como mensajes M209. Se observa que también en este caso la transmisión se lleva a cabo utilizando el WebSocket WS.

Después de finalizar el almacenamiento de los objetos B1, B2, B3, ..., el terminal 10 envía la notificación M211 al servidor 120 que ha finalizado el almacenamiento.

Al recibir la notificación M211, el servidor 120 envía (indicado por el mensaje M213) el objeto referente modificado AM al terminal 10 como una respuesta HTTP (ver la figura 3).

Para el almacenamiento, el terminal 10 guarda los objetos referidos B1, B2, B3, ... en un sistema de archivos en el terminal 10, tal como el almacenamiento temporal del navegador.

Cuando el objeto referente modificado AM se ha recibido en el terminal 10, los enlaces modificados ML1, ML2, ML3 apuntan a los objetos referidos B1, B2, B3, ... almacenados localmente en el sistema de archivos del terminal 10. En otras palabras, los enlaces modificados ML1, ML2, ML3 proporcionan referencias para que el navegador en el terminal 10 localice los objetos referidos B1, B2, B3, ... en el sistema de archivos local del terminal 10.

En la etapa P211, el terminal 10 recupera y proporciona los objetos referidos B1, B2, B3, ... en base al objeto referente modificado AM. Se observa que el resultado es que el contenido C solicitado por el terminal 10 con el mensaje M201 se reproduce en el terminal 10, pero de tal modo que el objeto referente A se recibe en el terminal 10 solamente después de que los objetos referidos B1, B2, B3, ... han sido cargados primero en el terminal 10 por medio del WebSocket WS.

Resumiendo, la figura 2 muestra el principio en mayor detalle: el servidor 120 recupera, además del objeto referente A que se puede recuperar opcionalmente, los objetos referidos B1, B2, B3, ... desde por lo menos un servidor remoto 140 utilizando el protocolo HTTP. A continuación, el servidor 120 transmite al terminal 10 los objetos referidos B1, B2, B3, ... sobre el WebSocket WS.

En la implementación de los inventores del servidor 120 y del procedimiento según la invención, todas las instrucciones del servidor 120 al cliente se envían en la respuesta M203, que contiene las instrucciones de

- 1) establecer un WebSocket WS que sigue al WebSocket estándar

Código JavaScript de ejemplo para el establecimiento del WebSocket:

```

45    var websocket = new
        WebSocket("ws://192.168.0.1:8888")

```

- 2) establecer el sistema de archivos que sigue el proyecto de norma de sistema de archivos.

Código JavaScript de ejemplo para establecer el sistema de archivos:

```
window.requestFileSystem(window.TEMPORARY, 1024*1024, function(filesystem)
```

3) recibir objetos web por medio del WebSocket WS y almacenarlos en la memoria asignada por el navegador

5 Ejemplo de pseudocódigo:

```
// cuando se recibe un objeto por medio del WebSocket WS, éste se escribe en el sistema de archivos websocket.onmessage = function (evt) {
```

```
// crear un objeto FileWriter para nuestro FileEntry (fileName).
```

```
fs.root.getFile(fileName, {create: true}, function(fileEntry) {
```

10 fileEntry.createWriter(function(fileWriter) {

```
fileWriter.onwriteend = function(event) {
```

```
debug("Write completed.");
```

```
};
```

```
var writer = new BlobBuilder();
```

15 writer.append(content);

```
fileWriter.write(writer.getBlob(type));
```

```
}, errorHandler);
```

```
);
```

20 4) cuando todos los objetos referidos B1, B2, B3, ... están almacenados en el sistema de archivos, el código genera la notificación M211, que sirve para informar al servidor 120 para que envíe el objeto referente modificado AM.

Código JavaScript de ejemplo para enviar la notificación M211 que es básicamente una solicitud HTTP agregando una cadena especial "//filesystem//":

```
window.location.href =
```

25 'http://www.example.com//filesystem//');

La funcionalidad de las instrucciones en la respuesta M203 es como el software cliente y todas las funciones necesarias son reinterpretadas en JavaScript.

30 En lugar de instalar software del lado del cliente en el sistema operativo del terminal 10, en la implementación de los inventores las funciones se instalan en directo. Dado que todos los objetos referidos B1, B2, B3, ... están ya dentro del navegador y los enlaces modificados ML1, ML2, ML3 proporcionan ya las indicaciones suficientes para que el navegador encuentre dónde están estos objetos referidos B1, B2, B3, ..., el objeto referente modificado AM tiene solamente que contener el objeto referente A (página HTML original) en la que los enlaces L1, L2, L3, ... han sido sustituidos con enlaces modificados ML1, ML2, ML3,...

35 No se debe entender que la invención se limita a las reivindicaciones de patente adjuntas, sino que se debe entender que abarca todos los equivalentes legales.



## REIVINDICACIONES

1. Un servidor (120) configurado para:
  - a) en respuesta a recibir una solicitud HTTP (M201) de contenido (C) desde un terminal (10), responder (P201) al terminal (10) con una respuesta HTML (M203) que contiene instrucciones configuradas para establecer un canal de comunicación bidireccional (WS) en el terminal (10) para la comunicación entre el servidor (120) y el terminal (10), como un WebSocket;
  - b) cargar (M205, M207) el contenido que comprende por lo menos un objeto referente (A) y una serie de objetos referidos (B1, B2, B3, ...) pertenecientes a dicho contenido (C), donde el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos (B1, B2, B3, ...) son objetos web referidos en dicha página de lenguaje de hipertexto;
  - c) generar un objeto referente modificado (AM) sustituyendo en el objeto referente (A) aquellos enlaces (L1, L2, L3, ...) a objetos referidos (B1, B2, B3, ...) que son enlaces externos, con enlaces modificados (ML1, ML2, ML3) que apuntan a una versión almacenada localmente del mismo objeto (B1, B2, B3, ...);
  - d) enviar los objetos referidos (B1, B2, B3, ...) al terminal (10) sobre el canal de comunicación bidireccional (WS); y
  - e) enviar el objeto referente modificado (AM) al terminal (10), de tal modo que los enlaces modificados (ML1, ML2, ML3) en el objeto referente modificado (AM) apuntan a los objetos referidos (B1, B2, B3, ...) ya enviados al terminal (10), solamente después de que el servidor (120) ha enviado los objetos referidos (B1, B2, B3, ...) al terminal (10).
2. Un servidor (120) según la reivindicación 1, en el que: el servidor (120) está configurado para enviar los objetos referidos (B1, B2, B3, ...) al terminal (10) en por lo menos un paquete.
3. Un servidor (120) según la reivindicación 2, en el que: el servidor (120) está configurado para enviar los objetos referidos (B1, B2, B3, ...) al terminal (10) en por lo menos un paquete, después de haber enviado instrucciones (M203) para establecer el WebSocket (WS) e instrucciones para guardar los objetos referidos (B1, B2, B3, ...) localmente en el terminal (10).
4. Un servidor (120) según cualquiera de las reivindicaciones anteriores, en el que: el servidor (120) está configurado para enviar el objeto referente modificado (AM) al terminal (10) como una respuesta HTTP.
5. Un procedimiento de funcionamiento de un servidor (120), que comprende las etapas de:
  - a) en respuesta a recibir una solicitud HTTP (M201) de contenido (C) desde un terminal (10), responder al terminal (10) con una respuesta HTML (M203) que contiene instrucciones configuradas para establecer un canal de comunicación bidireccional (WS) en el terminal (10) para la comunicación entre el servidor (120) y el terminal (10), como un WebSocket;
  - b) cargar (M205, M207) el contenido que comprende por lo menos un objeto referente (A) y una serie de objetos referidos (B1, B2, B3, ...) pertenecientes a dicho contenido (C), donde el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos (B1, B2, B3, ...) son objetos web referidos en dicha página de lenguaje de hipertexto;
  - c) generar un objeto referente modificado (AM) sustituyendo en el objeto referente (A) aquellos enlaces (L1, L2, L3, ...) a objetos referidos (B1, B2, B3, ...) que son enlaces externos, con enlaces modificados (ML1, ML2, ML3) que apuntan a una versión almacenada localmente del mismo objeto (B1, B2, B3, ...);
  - d) enviar al terminal (10) los objetos referidos (B1, B2, B3, ...) sobre el canal de comunicación bidireccional (WS); y
  - e) enviar al terminal (10) el objeto referente modificado (AM), de tal modo que los enlaces modificados (ML1, ML2, ML3) en el objeto referente modificado (AM) apuntan a los objetos referidos (B1, B2, B3, ...) ya enviados al terminal (10), solamente después de que se ha llevado a cabo la etapa d).
6. Un procedimiento según la reivindicación 5, en el que: los objetos referidos (B1, B2, B3, ...) se envían al terminal (10) en por lo menos un paquete.
7. Un procedimiento según la reivindicación 6, en el que: los objetos referidos (B1, B2, B3, ...) se envían al terminal (10) en por lo menos un paquete después de haber enviado instrucciones (M203) para establecer el WebSocket (WS) e instrucciones para guardar los objetos referidos (B1, B2, B3, ...) localmente en el terminal (10).
8. Un procedimiento según cualquiera de las reivindicaciones anteriores 5 a 7, en el que: el servidor (120) está configurado para enviar el objeto referente modificado (AM) al terminal (10) como una respuesta HTTP.

9. Un sistema que comprende por lo menos un servidor (120) según cualquiera de las reivindicaciones anteriores 1 a 4, y una serie de terminales (10) que son preferentemente terminales móviles, en el que: el sistema está configurado para utilizar dicho por lo menos un servidor (120) para responder a solicitudes HTTP (M201) de contenido que han sido enviadas por los terminales (10).

- 5 10. El sistema según la reivindicación 9, en el que: el servidor (120) responde a los terminales (10) con una respuesta HTML (M203) que contiene instrucciones configuradas para establecer un canal de comunicación bidireccional (WS) en el terminal (10) para una comunicación entre el servidor (120) y el terminal (10) como un WebSocket, carga (M205, M207) el contenido que comprende por lo menos un objeto referente (A) y una serie de objetos referidos (B1, B2, B3, ...) pertenecientes a dicho contenido (C), donde el objeto referente es una página de lenguaje de marcado de hipertexto y los objetos referidos (B1, B2, B3, ...) son objetos web a los que se hace referencia en dicha página de lenguaje de hipertexto, genera un objeto referente modificado (AM) sustituyendo en el objeto referente (A) aquellos enlaces (L1, L2, L3, ...) a los objetos referidos (B1, B2, B3, ...) que son enlaces externos, con enlaces modificados (ML1, ML2, ML3) que apuntan a una versión almacenada localmente del mismo objeto (B1, B2, B3, ...), envía los objetos referidos (B1, B2, B3, ...) al terminal (10) sobre el canal de comunicación bidireccional (WS), y envía el objeto referente modificado (AM) al terminal (10), de tal modo que los enlaces modificados (ML1, ML2, ML3) en el objeto referente modificado (AM) apuntan a los objetos referidos (B1, B2, B3) ya enviados al terminal (10), solamente después de que el servidor (120) haya enviado los objetos referidos (B1, B2, B3, ...) al terminal (10).

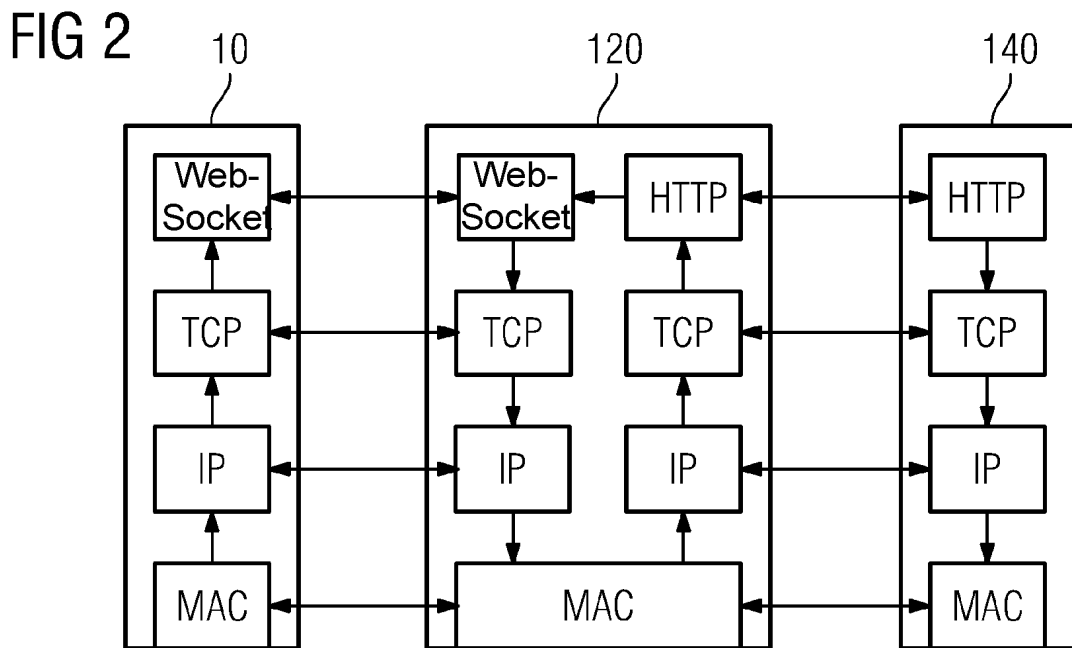
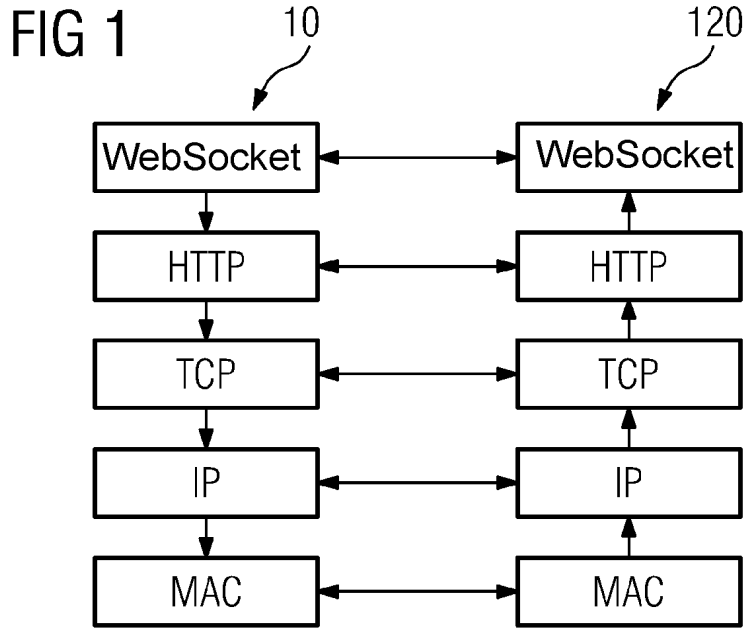


FIG 3

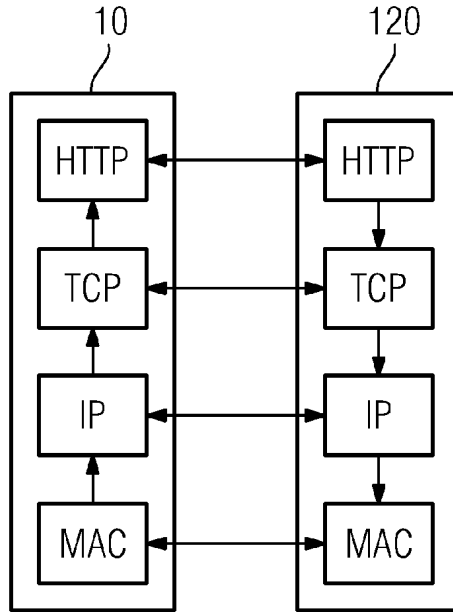


FIG 4

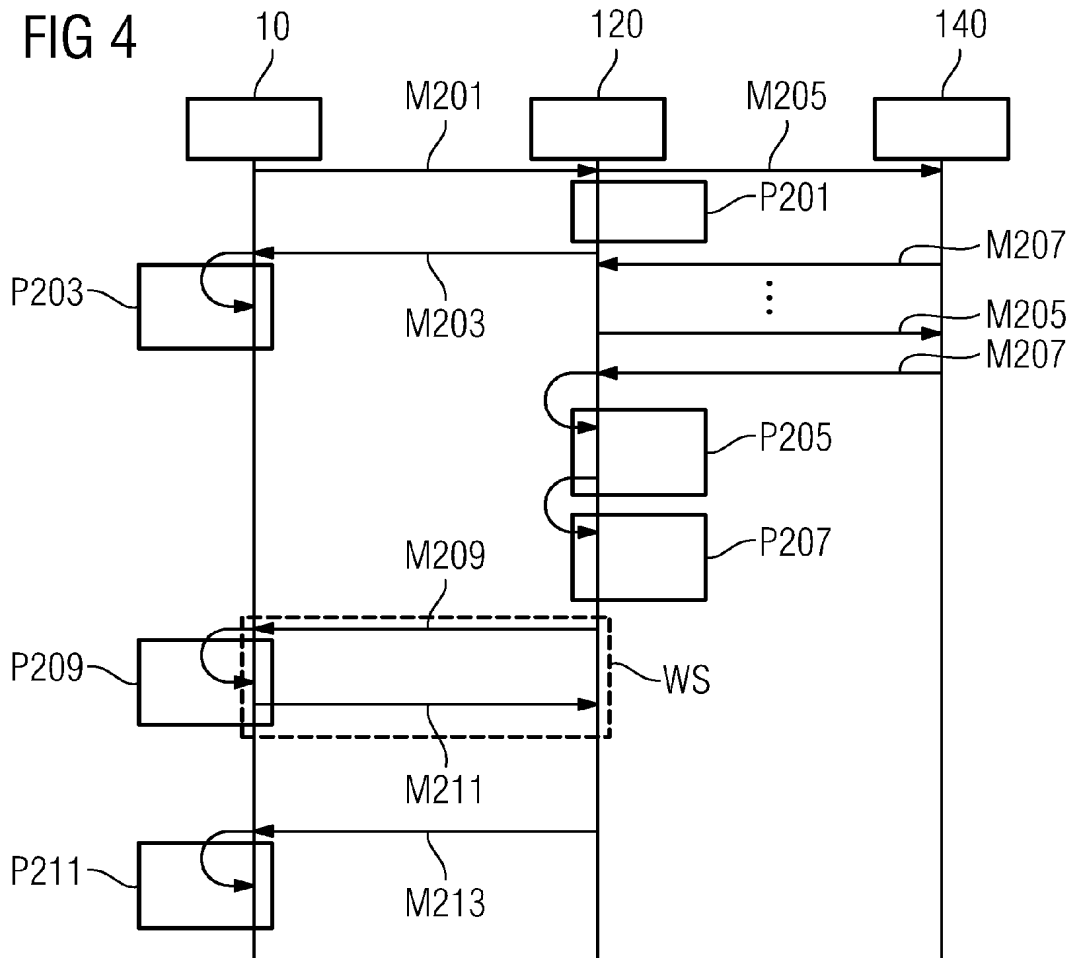


FIG 5

