(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 368 366**

(51) Int. CI.:
H04L 29/08 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Número de solicitud europea: **08767554 .2**
(96) Fecha de presentación: **05.05.2008**
(97) Número de publicación de la solicitud: **2145452**
(97) Fecha de publicación de la solicitud: **20.01.2010**

(54) Título: **MÉTODO Y SISTEMA PARA AMPLIAR LAS CAPACIDADES DE DISPOSITIVOS INTEGRADOS A TRAVÉS DE CLIENTES DE RED.**

(30) Prioridad:
07.05.2007 US 927978 P

(45) Fecha de publicación de la mención BOPI:
16.11.2011

(45) Fecha de la publicación del folleto de la patente:
16.11.2011

(73) Titular/es:
VORNE INDUSTRIES, INC.
1445 INDUSTRIAL DRIVE
ITASCA, IL 60143, US

(72) Inventor/es:
VORNE, Ramon, A.;
SAKS, Benjamin, D. y
TANG, Ke

(74) Agente: **Curell Aguila, Marcelino**

DESCRIPCIÓN

Método y sistema para ampliar las capacidades de dispositivos integrados a través de clientes de red.

Antecedentes

La presente invención se refiere a dispositivos integrados conectados a redes que tienen clientes tales como navegadores web que se ejecutan en uno o más ordenadores anfitriones.

Los dispositivos integrados (es decir, dispositivos que combinan software y hardware electrónicos y posiblemente piezas mecánicas u otros componentes, y que están diseñados específicamente para ejecutar una función o tarea dedicada; por ejemplo, máquinas expendedoras, electrodomésticos, controladores de motores, impresoras) se diseñan frecuentemente para funcionar en combinación con ordenadores anfitriones con el fin de proporcionar características tales como interfaces de usuario mejoradas (que usan la pantalla del ordenador anfitrión), acceso remoto (a través de una red a la cual están conectados tanto el ordenador anfitrión como el dispositivo integrado), y modernizaciones de microprogramas (cargando una versión nueva del microprograma en el dispositivo integrado desde el ordenador anfitrión). Aprovechando las capacidades y los recursos del ordenador anfitrión, el dispositivo integrado puede superar las restricciones de recursos internos que son inherentes debido a las limitaciones de coste y/o tamaño del dispositivo integrado. Estas restricciones se manifiestan frecuentemente como limitaciones en la cantidad de memoria (por ejemplo, bytes de memoria de acceso aleatorio) y/o el poder de procesado (por ejemplo, velocidad del procesador, tamaño del bus de datos, conjunto de instrucciones, y periféricos incorporados) del dispositivo integrado.

Centrándose más en la cuestión de las restricciones de memoria en los dispositivos integrados, resulta particularmente preocupante la memoria de acceso aleatorio (RAM). Típicamente, los microcontroladores de un solo chip, que con frecuencia se usan en dispositivos integrados, tienen una RAM limitada y se basan en otros tipos de memoria (por ejemplo, memoria flash) para almacenar programas y otros datos constantes. Por ejemplo, una plataforma de microprocesador actualmente popular para dispositivos integrados es la ARM7. Dos de los proveedores líderes de ARM7, Atmel Corporation y NXP Semiconductors, ofrecen ambos, dispositivos ARM7 con conectividad de red (en forma de controladores de acceso a los medios de Ethernet). La familia AT91SAM7X de Atmel proporciona una memoria flash que es cuatro veces la cantidad de memoria RAM (por ejemplo, el AT91SAM7XC512 de gama alta incluye una flash de 512 KB y una RAM de 128 KB). La diferencia es aún más pronunciada en el NXP LPC2368, que incluye 512 KB de flash y solamente 58 KB de RAM. Puesto que la RAM es frecuentemente un recurso muy limitado en dispositivos integrados, resulta especialmente deseable reducir el uso de RAM en dispositivos integrados.

Dos técnicas comunes para aprovechar las capacidades y los recursos de los ordenadores anfitriones son: i) instalar software personalizado en cada ordenador anfitrión que interaccionará con el dispositivo integrado, o ii) incorporar un servidor HTTP en el dispositivo integrado que genere contenido adecuado para un cliente HTTP (es decir, un navegador web) en el ordenador anfitrión. Cada método tiene sus puntos fuertes y sus puntos débiles.

Un punto fuerte del software personalizado es que permite que los dispositivos integrados, limitados en cuanto a recursos, aprovechen exhaustivamente las capacidades y los recursos del ordenador anfitrión, debido a la capacidad del software personalizado de acceder a y controlar muchos aspectos del ordenador anfitrión.

Un punto débil del software personalizado es que típicamente necesita ser instalado y mantenido en cada ordenador anfitrión que accederá al dispositivo integrado. En la práctica, esto frecuentemente resulta farragoso, consume mucho tiempo, y es caro, especialmente en entornos comerciales en los que típicamente solo a los departamentos de IT se les permite instalar software. Cada versión nueva del software personalizado requiere actualizaciones o instalaciones nuevas, y con frecuencia surgen problemas de compatibilidad debido a interacciones entre el software personalizado y versiones diferentes de sistemas operativos del ordenador, combinaciones diferentes de otras aplicaciones de software personalizado instaladas en el ordenador anfitrión, y/o desadaptaciones entre versiones del software personalizado y versiones de los dispositivos integrados.

Un punto fuerte de incorporar un servidor HTTP en el dispositivo integrado es que proporciona un cliente que ocupa un “espacio cero”, lo cual significa que se puede usar un cliente HTTP convencional (por ejemplo, un navegador web) de un ordenador anfitrión para acceder al dispositivo integrado desde cualquier ordenador anfitrión que tenga instalado el cliente. Puesto que la gran mayoría de ordenadores personales tienen navegadores web pre-instalados, esto constituye una mejora principal con respecto al software personalizado.

Un punto débil de la incorporación de un servidor HTTP en el dispositivo integrado es que las restricciones de recursos del dispositivo integrado, tales como las limitaciones antes mencionadas de memoria y poder de procesado, pueden tener un impacto importante en la experiencia del usuario en términos de i) calidad, tal como la facilidad de utilización de una interfaz de usuario o la sofisticación de un informe que pueda ser generado, ii) cantidad, tal como el tamaño de un informe que pueda ser generado o el tamaño de un archivo que pueda ser leído,

iii) sensibilidad, tal como la rapidez con la que el dispositivo integrado puede generar contenido solicitado, y/o iv) escalabilidad, tal como el número de clientes a los que se puede prestar servicio simultáneamente.

Aunque existen tecnologías disponibles que proporcionan grados variables de procesado del lado del cliente (por ejemplo, el Reproductor Flash® de Adobe Systems Incorporated, OpenLaszlo™ de Laszlo Systems Incorporated, y el Entorno en Tiempo de Ejecución Java™ de Sun Microsystems Incorporated), estas tecnologías típicamente no están diseñadas u optimizadas de manera específica para trabajar con dispositivos integrados, y, por lo tanto, típicamente no tienen en cuenta los requisitos y limitaciones especiales de los dispositivos integrados de recursos restringidos. Como consecuencia, las tecnologías existentes padecen en general uno o más de los siguientes problemas:

1. No están diseñadas para minimizar explícitamente el uso de memoria (por ejemplo, RAM y/o flash) y/o el ancho de banda de procesado en el servidor (es decir, dispositivo integrado) y, por lo tanto, pueden no ejecutarse de manera eficaz en dispositivos integrados de recursos restringidos.
2. No proporcionan herramientas para leer, escribir, y/o manipular de manera arbitraria archivos grandes cuando se tienen en cuenta los recursos limitados de los dispositivos integrados.
3. No están diseñadas para actualizar dinámicamente contenido o lo hacen de manera ineficaz.
4. No separan limpiamente el contenido estático (que puede ser almacenado en memoria caché por el cliente) del contenido dinámico.
5. No están diseñadas para un procesado general en el lado del cliente (por ejemplo, se centran en el procesado de la capa de presentación).
6. Requieren herramientas de desarrollo privativas (por ejemplo, disponibles solamente en una empresa en particular, y/o solamente para una plataforma en particular) limitando de este modo las opciones de desarrollo.
7. Requieren componentes del lado del servidor, lenguajes de programación y/o lenguajes de guión de instrucciones (*scripting*) que en general no están disponibles para los dispositivos integrados o que resultan adecuados de manera deficiente para su uso en dichos dispositivos.
8. Requieren la instalación de software adicional en el cliente (por ejemplo, módulos *plug-in* del navegador).
9. No soportan una amplia gama de clientes (por ejemplo, una amplia gama de plataformas de navegador).
10. Soportan solamente un tipo único de contenido (por ejemplo, archivos SWF Flash®) o una gama limitada de contenido.
11. No proporcionan herramientas para acceder a recursos desde dominios externos.

Debe observarse que la lista anterior de problemas se proporciona a título ilustrativo, y no está destinada a servir como una lista exhaustiva de cada aspecto de las tecnologías existentes que las pueda hacer inapropiadas para dispositivos integrados.

Por lo tanto, se requiere un método eficaz para que dispositivos integrados de recursos limitados interaccionen con ordenadores anfitriones, el cual pueda proporcionar idealmente uno o más de los siguientes puntos:

1. Un cliente que ocupe un espacio cero y que no requiera la instalación de ningún software personalizado en el ordenador anfitrión.
2. Una reducción significativa en la cantidad de memoria y el poder de procesado que requiere el dispositivo integrado para producir contenido sofisticado, complejo, y de alta calidad, incluyendo contenido dinámico.
3. Una capacidad de generar contenido que sea mucho más grande (potencialmente, órdenes de magnitud más grande) de lo que puede caber dentro de la memoria disponible del dispositivo integrado.
4. Una capacidad de almacenar archivos arbitrariamente grandes en el ordenador anfitrión y/o en sistemas de archivos accesibles para el ordenador anfitrión.
5. Una capacidad de leer, procesar, y extraer información de archivos arbitrariamente grandes del ordenador anfitrión y/o de sistemas de archivos accesibles para el ordenador anfitrión.
6. Una solución que maximice la cantidad de contenido que puede almacenar en memoria caché el cliente.

7. Una solución generalizada y “genérica” que no esté limitada significativamente en el tipo de contenido que se puede generar o el tipo de procesado que se puede realizar en el cliente.

8. Una solución que no dependa de productos específicos de terceros, módulos plug-in de navegadores, herramientas de desarrollo, etcétera.

9. Una solución que sea eficaz sobre una gama amplia de clientes y dispositivos integrados.

10. Una solución que se pueda adaptar de manera sencilla y flexible a los requisitos de la aplicación específica y a los recursos del dispositivo integrado específico.

11. Una capacidad de que el cliente acumule de manera eficaz datos de múltiples dispositivos integrados al mismo tiempo que planteando unos requisitos mínimos de memoria y procesado sobre los dispositivos integrados.

12. Una experiencia global del usuario en general mejorada cuando se interacciona con el dispositivo integrado a través del ordenador anfitrión.

Sumario

En la presente memoria se da a conocer un MÉTODO Y SISTEMA PARA AMPLIAR LAS CAPACIDADES DE DISPOSITIVOS INTEGRADOS, A TRAVÉS DE CLIENTES DE RED, que aprovecha los recursos de memoria y procesado de clientes, tales como navegadores (“clientes”) web, que se ejecutan en uno o más ordenadores anfitriones.

El dispositivo integrado de recursos limitados, conectado en red (“dispositivo integrado”) actúa como un simple servidor de archivos y datos para clientes, y estos clientes asumen la responsabilidad de la generación de contenido y otras tareas computacionales. Puesto que los clientes que se ejecutan en ordenadores anfitriones en general tienen acceso a una memoria y un poder de procesado mayores, en órdenes de magnitud, que los dispositivos integrados típicos, en general son capaces de generar un contenido bastante más enriquecido y realizar tareas computacionales bastante más sofisticadas que los propios dispositivos integrados.

Además, múltiples clientes pueden procesar y manipular simultáneamente archivos y datos servidos desde un único dispositivo integrado. El resultado final es un sistema altamente escalable que maximiza el número de clientes que puede soportar simultáneamente un dispositivo integrado, y que mejora significativamente la calidad (y cantidad) de contenido que se puede generar y la sofisticación de las tareas computacionales que se pueden realizar.

Frecuentemente, puede existir una necesidad de almacenar o abrir este contenido generado, en el ordenador anfitrión, pero puede que el cliente no disponga de los medios para lograr esto, típicamente debido a restricciones de seguridad (por ejemplo, las restricciones impuestas sobre navegadores web usados comúnmente, para escribir archivos en el ordenador anfitrión y/o en sistemas de archivos accesibles para el ordenador anfitrión). Varias formas de realización de la invención incluyen también un método y un sistema para leer y escribir archivos arbitrariamente grandes desde y en el ordenador anfitrión y/o sistemas de archivos accesibles para el ordenador anfitrión (a lo cual se hace referencia como “rebote de archivos”) al mismo tiempo que superando las limitaciones de memoria y procesado del dispositivo integrado, así como superando las limitaciones impuestas por restricciones de seguridad del cliente.

El sistema, según varias formas de realización, comprende cuatro elementos principales: un motor de procesado del cliente, archivos de plantillas estáticas, conjuntos de datos dinámicos, y canales de comunicación gestionados. Estos elementos se describen de forma más detallada posteriormente.

El motor de procesado del cliente es responsable de coordinar trabajo realizado en el cliente en nombre del dispositivo integrado. Desde la perspectiva del dispositivo integrado, el motor de procesado del cliente es simplemente un recurso estático (o una colección de recursos estáticos) almacenado en el dispositivo integrado y transmitido al cliente bajo demanda.

No obstante, desde la perspectiva del cliente, una vez que el motor de procesado del cliente se ha cargado en el cliente, el mismo es un programa ejecutable (por ejemplo, un programa JavaScript) responsable de realizar el procesado del lado del cliente. El motor de procesado del cliente interpreta archivos de plantillas estáticas y lleva a cabo las instrucciones especificadas en los mismos. El motor de procesado del cliente está destinado y diseñado explícitamente para minimizar recursos y procesado requeridos dentro del dispositivo integrado y para transferir trabajo desde el dispositivo integrado al cliente. El motor de procesado del cliente en esencia se convierte en un “sistema operativo” que se ejecuta dentro del cliente, y eleva al cliente de ser un medio de entrega de contenido a ser un “nodo” de procesado en un sistema informático distribuido.

Los archivos de plantillas estáticas son responsables de proporcionar información necesaria para realizar tareas de procesado específicas (por ejemplo, generar un informe, reproducir una página web, etcétera). Desde la perspectiva del dispositivo integrado, un archivo de plantillas estáticas es simplemente un recurso estático transmitido al cliente bajo demanda. Desde la perspectiva del cliente, un archivo de plantillas estáticas contiene un conjunto de instrucciones de procesado para el motor de procesado del cliente. En esencia, el dispositivo integrado delega tareas de procesado contenidas por archivos de plantillas estáticas al cliente, con el fin de minimizar recursos y procesado requeridos dentro del dispositivo integrado y con el fin de transferir trabajo desde el dispositivo integrado al cliente.

Los conjuntos de datos dinámicos son colecciones de datos que se intercambian entre el dispositivo integrado y el cliente. A diferencia de los archivos de plantillas estáticas, que son recursos estáticos, cada conjunto de datos dinámicos se genera dinámicamente por medio del dispositivo integrado (o alternativamente por medio del cliente). La Notación de Objetos de JavaScript (JSON), descrita mediante la RFC 4627, resulta particularmente útil como formato de conjunto de datos dinámicos ya que resulta muy sencilla de analizar sintácticamente y de generar, y se diseñó específicamente para proporcionar una representación de datos compacta. Los conjuntos de datos dinámicos, en esencia, encapsulan el aspecto “dinámico” del contenido con una representación compacta de datos, y, conjuntamente con una variedad de técnicas que minimizan los recursos y el procesado requeridos dentro del dispositivo integrado, transfieren además trabajo desde el dispositivo integrado al cliente.

Los canales de comunicación gestionados son enlaces de comunicación bidireccionales entre el motor de procesado del cliente y un dispositivo integrado. El término “gestionado” se refiere al hecho de que los canales de comunicación son controlados por el motor de procesado del cliente, no directamente por el cliente como se produce tradicionalmente. El motor de procesado del cliente puede usar canales de comunicación gestionados, para mantener la comunicación en curso con uno o más dispositivos integrados de una manera ininterrumpida que resulta invisible para el usuario. Los canales de comunicación gestionados se pueden implementar con una variedad de técnicas, tales como la XHR (XMLHttpRequest) y la manipulación de documentos. Con la manipulación de documentos, el motor de procesado del cliente modifica contenido existente (por ejemplo, HTML, XHTML, etcétera), lo cual provoca que el cliente realice una solicitud de comunicación al dispositivo integrado. La manipulación de documentos se logra a través de técnicas tales como el método document.write () de JavaScript, modificaciones en el Modelo de Objeto de Documento, y modificaciones en la propiedad innerHTML del documento.

El motor de procesado del cliente analiza sintácticamente cada archivo de plantillas estáticas y se comunica con el dispositivo integrado usando uno o más canales de comunicación gestionados, para solicitar, recibir, y/o presentar conjuntos de datos dinámicos, que son usados por el motor de procesado del cliente conjuntamente con archivos de plantillas estáticas para generar, procesar, transformar, manipular, y/o acumular contenido así como realizar otras tareas computacionales.

El rebote de archivos se puede lograr usando, por ejemplo, el protocolo HTTP, para enviar archivos al dispositivo integrado en una serie de uno o más paquetes que el dispositivo integrado simplemente hace “rebotar” de vuelta al cliente. El dispositivo integrado únicamente conserva un paquete dado durante el tiempo que se tarda en hacer rebotar ese paquete hacia el cliente, después de lo cual el paquete se puede descartar. Los paquetes pueden tener un tamaño arbitrario, y el dispositivo integrado únicamente necesita reservar recursos de memoria suficientes para almacenar en memoria intermedia un único paquete cada vez (es decir, el dispositivo integrado no necesita almacenar el archivo completo o ni siquiera una parte significativa del mismo).

Los archivos que se leen desde el ordenador anfitrión y/o desde sistemas de archivos accesibles para el ordenador anfitrión (por ejemplo, usando la carga de archivos basada en formularios HTML) haciendo uso del rebote de archivos están totalmente disponibles para el motor de procesado del cliente, el cual puede manipular, transformar, y/o acceder de manera selectiva a su contenido como un proxy para el dispositivo integrado.

Los archivos que se escriben en el ordenador anfitrión y/o en sistemas de archivos accesibles para el ordenador anfitrión (por ejemplo, usando el encabezamiento Disposición de Contenido de HTTP) haciendo uso del rebote de archivos se generan típicamente por medio del motor de procesado del cliente a partir de archivos de plantillas estáticas y/o conjuntos de datos dinámicos. Los conjuntos de datos dinámicos se pueden enviar por flujo continuo desde el dispositivo integrado al cliente a medida que se producen los datos subyacentes, haciendo que, desde la perspectiva del dispositivo integrado, esto resulte transitorio. Mientras se generan archivos, el motor de procesado del cliente puede manipular y/o transformar contenido en nombre del dispositivo integrado.

El rebote de archivos puede reducir drásticamente los recursos requeridos por un dispositivo integrado para leer, acceder a, generar, y escribir archivos arbitrariamente grandes desde y en el ordenador anfitrión.

Estos y otros aspectos de la invención se pondrán más claramente de manifiesto a partir de la siguiente descripción y los dibujos adjuntos.

Breve descripción de los dibujos

La invención se describe haciendo referencia a varias formas de realización que se ilustran en los dibujos y siguiendo la descripción detallada proporcionada a continuación.

La FIG. 1 es un diagrama de bloques que muestra una vista de nivel superior de dispositivos integrados conectados en red y ordenadores anfitriones con clientes.

La FIG. 2 es un diagrama de bloques que muestra relaciones entre elementos principales del sistema.

La FIG. 3 es un diagrama de bloques que muestra subcomponentes del motor de procesado del cliente.

La FIG. 4 es un ejemplo de un conjunto de datos dinámicos que contiene datos de tiempo real generados por un dispositivo integrado.

La FIG. 5 es un ejemplo de un archivo de plantillas estáticas abstractas.

La FIG. 6 es un ejemplo de contenido generado por el motor de procesado del cliente a partir del archivo de plantillas estáticas abstractas de la FIG. 5 y el conjunto de datos dinámicos de la FIG. 4.

La FIG. 7 muestra el contenido de la FIG. 6 según es reproducido por un navegador web.

La FIG. 8 es un ejemplo de un archivo de plantillas estáticas literales para generar un documento de Formato de Texto Enriquecido.

La FIG. 9 es el contenido generado por el motor de procesado del cliente a partir del archivo de plantillas estáticas literales de la FIG. 8 y el conjunto de datos dinámicos de la FIG. 4.

La FIG. 10 muestra el contenido de la FIG. 9 según es reproducido por un procesador de texto.

La FIG. 11 es un diagrama de secuencias que muestra etapas para generar contenido con elementos tanto estáticos como dinámicos.

La FIG. 12 es un diagrama de secuencias que muestra etapas para usar el rebote de archivos con el fin de almacenar en el ordenador anfitrión contenido generado en el cliente.

La FIG. 13 es un diagrama de secuencias que muestra etapas para usar el rebote de archivos con el fin de cargar en el cliente un archivo accesible para el ordenador anfitrión (pero no directamente accesible para el cliente).

Descripción detallada

Aunque la presente invención puede materializarse en formas de realización de muchas maneras diferentes, en los dibujos se muestra, y en la presente memoria se describirá detalladamente, una forma de realización preferida de la invención, entendiendo que la presente exposición debe considerarse como una ejemplificación de los fundamentos de la invención y no pretende limitar el amplio aspecto de la invención a formas de realización ilustradas.

Varias formas de realización de la invención proporcionan un método y un sistema para permitir que dispositivos integrados de recursos limitados, conectados en red (a los que, a continuación, se hará referencia simplemente como “dispositivos integrados”) aprovechen los recursos de memoria y procesado de clientes, tales como navegadores web (a los que, en lo sucesivo, se hará referencia como “clientes”), que están instalados en dispositivos computacionales anfitriones, tales como ordenadores personales o clientes ligeros (a los que, en lo sucesivo, se hará referencia como “ordenadores anfitriones”), de tal manera que se superen restricciones de recursos internos de los dispositivos integrados. En la FIG. 1 puede verse una vista de nivel superior, que muestra una pluralidad de dispositivos integrados 102, y una pluralidad de clientes 104 en ordenadores anfitriones 106, conectados a una red de comunicaciones compartida (es decir, común) 108.

La capacidad de usar un navegador web convencional como cliente 104 para interaccionar con el dispositivo integrado 102 hace que la solución ocupe un “espacio cero” desde la perspectiva del ordenador anfitrión 106 – no es necesario instalar ningún software personalizado puesto que la gran mayoría de ordenadores anfitriones tienen dichos clientes preinstalados.

Para minimizar el procesado realizado por el dispositivo integrado 102, el dispositivo integrado adopta la función de un simple servidor de archivos y datos que entrega dos tipos básicos de información, tal como puede observarse en la FIG. 2:

- Archivos de recursos estáticos 202 tales como JavaScript y XML

- Conjuntos de datos dinámicos 204 tales como datos de tiempo real, en un formato compacto tal como JSON (Notación de Objetos de JavaScript)

Para ser considerado como “archivo” a efectos de esta descripción, no es necesario que un recurso estático esté almacenado en un sistema de archivos tradicional. Por ejemplo, un recurso estático se podría almacenar como un objeto de algún tipo (por ejemplo, una secuencia de bytes) en memoria flash.

En el caso de archivos de recursos estáticos 202, el dispositivo integrado 102 simplemente transmite estos archivos al cliente 104 cada vez que los mismos son solicitados. No realiza ningún procesado significativo sobre los archivos más allá de la transmisión. Los archivos de recursos estáticos se pueden almacenar en memoria flash del dispositivo integrado, y se pueden transmitir como una secuencia de “paquetes” pequeños hacia el cliente, minimizando de este modo el uso de recursos de la RAM.

En el caso de conjuntos de datos dinámicos 204, el dispositivo integrado 102 transmite datos en un formato sencillo y compacto, tal como el formato JSON mostrado en la FIG. 4.

Haciendo referencia a la FIG. 2, el procesado de conjuntos de datos dinámicos 204 por parte del dispositivo integrado 102 se puede minimizar representando cada elemento de datos de una manera que sea muy parecida a su representación interna nativa en el dispositivo integrado. Por ejemplo, en el conjunto de datos dinámicos de la FIG. 4, el elemento de datos fecha/hora 404 tiene un valor de 202809600, lo cual representa el número de segundos desde el cambio de siglo (segundos desde 1/1/2000 12:00 AM), que, en este ejemplo, es también la representación interna de la fecha/hora del dispositivo integrado. El motor de procesado del cliente puede ser capaz de dar formato a este elemento de datos de muchas maneras diferentes, tales como 5/6/2006, 2006-06-05T08:00:00, ó 5 de junio de 2006 8:00 AM. Por ejemplo, el motor de procesado de cliente da formato a la fecha/hora 404 como 5/6/2006 8:00 AM dentro del subtítulo 704 (FIG. 7) y el subtítulo 1002 (FIG. 10).

Haciendo referencia a la FIG. 2, el dispositivo integrado 102 se centra en servir archivos y datos sin procesar en lugar de centrarse en la generación de contenido final (por ejemplo, páginas web XHTML, informes de Formato de Texto Enriquecido, etcétera). El motor de procesado del cliente 206 es responsable de transformar los archivos de plantillas estáticas 208 y los conjuntos de datos dinámicos 204 que recibe desde el dispositivo integrado para producir el contenido generado 212. Esta división de responsabilidades se aprovecha del hecho de que los ordenadores anfitriones 106, y los clientes 104 que se ejecutan en ellos, típicamente tienen unos recursos de memoria y de procesado bastante mayores que los dispositivos integrados, con frecuencia multiplicados por órdenes de magnitud.

De este modo, adoptando la función de un servidor básico de archivos y datos tal como se ha descrito anteriormente, el dispositivo integrado puede prestar servicio a muchos clientes con pocos recursos. Adicionalmente, múltiples clientes pueden procesar y manipular simultáneamente archivos y datos servidos desde un único dispositivo integrado. El resultado final es un sistema altamente escalable que maximiza el número de clientes que puede soportar simultáneamente un dispositivo integrado, y que mejora significativamente la calidad de contenido que se puede generar.

Es importante enfatizar que el dispositivo integrado 102 no es responsable de la generación de contenido. No genera interfaces de usuario, interfaces de diagnóstico, pantallas de información, informes, archivos de configuración, etcétera. Esa responsabilidad queda delega al cliente 104 a través del motor de procesado del cliente 206. Los archivos y datos servidos por el dispositivo integrado son procesados, transformados, manipulados, y combinados por el motor de procesado del cliente para generar varias formas de contenido 212. Y, tal como se ilustra en la FIG. 2, el motor de procesado del cliente 206 puede ser almacenado en memoria caché por el cliente 104, de modo que únicamente es necesario cargarlo desde el dispositivo integrado una vez por cliente.

Es también importante enfatizar que, aunque esta descripción se centra en la generación de contenido del lado del cliente, al cliente también se le pueden delegar otras tareas de procesado (por ejemplo, procesado intensivo (*crunching*) de números, análisis de datos, y validación de datos).

Existen cuatro elementos principales que interaccionan y que son compartidos entre el dispositivo integrado 102 y el cliente 104: archivos de plantillas estáticas 208, conjuntos de datos dinámicos 204, motor de procesado del cliente 206, y canales de comunicación gestionados 210.

Archivos de plantillas estáticas

El primer elemento es el archivo de plantillas estáticas 208. Cada archivo de plantillas estáticas se almacena en el dispositivo integrado 102 (por ejemplo, en memoria flash) y se transfiere al cliente 104 según sea necesario. Puesto que cada archivo de plantillas estáticas es un archivo de recursos estáticos, el dispositivo integrado simplemente lo lee de su memoria y lo transmite al cliente. No es necesario ningún procesado adicional significativo en el dispositivo

integrado. Típicamente, habrá una pluralidad de archivos de plantillas estáticas almacenados en el dispositivo integrado.

Desde la perspectiva del cliente 104, un archivo de plantillas estáticas 208 proporciona un conjunto de instrucciones de procesado. Por ejemplo, un archivo de plantillas estáticas puede contener instrucciones que se pueden usar para generar una página web, o un informe, o puede contener instrucciones computacionales generales. Los archivos de plantillas estáticas se escriben generalmente de una manera sencilla y compacta que pueda ser analizada sintácticamente de forma fácil para generar contenido o realizar otro procesado. El XML es un formato lógico para archivos de plantillas estáticas, ya que los documentos XML pueden resultar tan sencillos o complejos como se necesite para una aplicación particular, y el XML es analizado sintácticamente de manera fácil por clientes tales como navegadores web.

La FIG. 5 muestra un ejemplo de un archivo de plantillas estáticas que se podría usar para generar una página web, la FIG. 6 muestra contenido XHTML correspondiente generado por el motor de procesado del cliente, y la FIG. 7 muestra la página web generada, tal como aparece en un navegador web. De modo similar, la FIG. 8 muestra un ejemplo de un archivo de plantillas estáticas usado específicamente para generar un informe de Formato de Texto Enriquecido, la FIG. 9 muestra contenido correspondiente de Formato de Texto Enriquecido generado por el motor de procesado del cliente, y la FIG. 10 muestra el informe generado tal como aparece en un procesador de texto. En ambos casos, la fuente de datos dinámicos es el conjunto de datos dinámicos de la FIG. 4.

Los archivos de plantillas estáticas pueden proporcionar instrucciones de procesado que son abstractas (es decir, que describen el resultado final deseado sin proporcionar instrucciones explícitas sobre cómo lograrlo) y/o literales (es decir, que describen explícitamente cómo lograr el resultado final deseado).

El archivo de plantillas estáticas mostrado en la FIG. 5 contiene instrucciones abstractas que se pueden usar para generar la página web mostrada en la FIG. 7. Algunos de los elementos y atributos clave son los siguientes:

- El atributo de etiqueta 504 (FIG. 5) se usa para proporcionar etiquetas de texto para muchas partes de la página web mostrada en la FIG. 7, incluyendo el título 702, el subtítulo 704, el recuadro de variables 706 (cada recuadro de variables agrupa una colección de pares relacionados de nombre/valor), y la variable 708 (cada variable representa un único par nombre/valor).

- Haciendo referencia de nuevo a la FIG. 5, el atributo de índice 502 identifica un elemento de datos de un conjunto de datos dinámicos asociado a insertar en la página web (por ejemplo, como la parte de valor de un par de nombre/valor). Este atributo es solamente necesario si el orden en el que se usan los elementos de datos del conjunto de datos dinámicos dentro del archivo de plantillas estáticas es diferente de su orden en el conjunto de datos dinámicos. Por ejemplo, el atributo de índice 502 en el elemento de título, que tiene un valor de índice de “0”, identifica y estaría asociado al elemento de datos “Línea de Ensamblaje 12” 402 (FIG. 4) del conjunto de datos dinámicos. Debe observarse que el índice tiene base en cero.

- El atributo de formato 506 se usa para proporcionar instrucciones de formateo para elementos de datos. En este ejemplo, la convención de instrucciones de formateo (por ejemplo, “#0.00%”) es muy similar a los códigos de formato personalizado usados en el producto de hojas de cálculo Excel® de Microsoft Corporation.

- El elemento de pie de página 508 es una instrucción abstracta usada para indicar que se debería dibujar un pie de página en la parte inferior de la misma. Un identificador de pie de página individual resulta adecuado para aplicaciones que usan un pie de página uniforme en parte o en la totalidad de las páginas, en cuyo caso la generación completa del pie de página la puede realizar el motor de procesado del cliente. Esto ilustra otra ventaja del uso de archivos de plantillas estáticas – las instrucciones de procesado abstractas pueden hacer que resulte más sencillo diseñar el contenido, y el motor de procesado del cliente se puede usar para proporcionar un grado de uniformidad en el aspecto y la funcionalidad del contenido generado. Un ejemplo de funcionalidad generada por un motor de procesado del cliente es el hiperenlace 602 (FIG. 6) incluido en el pie de página generado.

- Los elementos de recuadro_variables 512 y panel_apilamiento 510 son otros ejemplos de instrucciones abstractas. Un recuadro_variables describe un cierto tipo de objeto de presentación (en este caso, un grupo de pares nombre/valor relacionados, donde cada par nombre/valor es un elemento variable), y un panel_apilamiento describe cómo distribuir objetos de presentación (en este caso horizontalmente).

El archivo de plantillas estáticas mostrado en la FIG. 8 usa instrucciones literales para describir el informe de Formato de Texto Enriquecido (RTF) mostrado en la FIG. 10. El archivo ejemplificativo de plantillas estáticas de la FIG. 8 soporta un formato de informe que consta de contenido de elemento de encabezamiento 802 que se sitúa en el comienzo del informe, contenido de elemento de cuerpo_registro 808 que se repite para cada “registro de datos”, contenido de elemento de separador_registros 806 que se sitúa entre cada registro de datos, y contenido de elemento de pie de página 804 que se sitúa en el mismo final del informe. De este modo, el archivo ejemplificativo de plantillas estáticas de la FIG. 8 se podría usar para generar un informe de una sola página o un informe de 1.000

páginas. Por brevedad, la FIG. 9 y la FIG. 10 representan un informe de una sola página generado a partir del registro de datos individual mostrado en el conjunto de datos dinámicos de la FIG. 4. La mayor parte del archivo ejemplificativo de plantillas estáticas de la FIG. 8 es contenido RTF literal que se copiará en el informe. Algunos de los elementos y atributos clave son los siguientes:

- El elemento de encabezamiento 802 y el elemento de pie de página 804 contienen texto literal que se situará en el comienzo y el final, respectivamente, del informe generado. De manera similar, el elemento de separador_registros 806 contiene texto literal que se situará entre registros individuales (pero antes del primer registro y no después del último registro).

- El elemento de cuerpo_registro 808 contiene principalmente texto literal que se copia directamente en el informe, eliminando cualquier espacio en blanco anterior y posterior.

- El elemento de definición 810 y el elemento de inserción 812 proporcionan un mecanismo sencillo usado para evitar la repetición de texto literal, y de este modo reducen el tamaño de un archivo de plantillas estáticas. El elemento de definición captura un bloque de texto literal en su punto de primer uso de manera que se pueda repetir (es decir, insertar) posteriormente con un elemento de inserción.

- El elemento de variable 814 marca cada lugar en el que se van a insertar datos dinámicos con formato en el informe. Debe observarse que no se requiere un atributo de etiqueta 504 (FIG. 5) puesto que la etiqueta se incluye como parte del texto literal, y no se requiere un atributo de índice 502 (FIG. 5) ya que el orden de los elementos de datos del elemento de variable es el mismo que su orden en el conjunto de datos dinámicos. Debe observarse también que el atributo de formato 506 (FIG. 8) tiene el mismo comportamiento que el atributo de formato 506 (FIG. 5).

Un archivo de plantillas estáticas podría incluir referencias a otros archivos de plantillas estáticas (o incluso a sí mismo), permitiendo de este modo un planteamiento “modular” para la construcción de conjuntos de instrucciones de procesado para el motor de procesado del cliente. Como archivos de recursos estáticos, en el dispositivo integrado se pueden almacenar archivos de plantilla estáticas en forma comprimida (por ejemplo, el formato zip de GNU, al que se hace referencia también como formato gzip), lo cual reduce el espacio ocupado por la memoria en el dispositivo integrado. Los mismos también se pueden transferir al cliente en forma comprimida (por ejemplo, formato gzip) para reducir el tiempo de transferencia y conservar recursos. El espacio ocupado por la memoria en el dispositivo integrado se puede reducir todavía más cargando archivos de plantillas estáticas desde una o más fuentes que no sean el dispositivo integrado (por ejemplo, desde un anfitrión externo).

Los archivos de plantillas estáticas también se pueden crear y/o modificar en el cliente. Por ejemplo, además de analizar sintácticamente archivos de plantillas estáticas, el motor de procesado del cliente puede contener lógica para generarlos (típicamente de manera conjunta con información recopilada a través de una interfaz de usuario). Los archivos de plantillas estáticas creados y/o modificados de tal manera se pueden almacenar dentro del dispositivo integrado y/o se pueden interpretar inmediatamente por medio del motor de procesado del cliente (por ejemplo, para una ejecución “de una sola vez” o para proporcionar una característica de vista previa). Los archivos de plantillas estáticas creados y/o modificados de tal manera, que a continuación se almacenan dentro del dispositivo integrado, siguen siendo “estáticos” desde la perspectiva tanto del dispositivo integrado (que no necesita realizar ningún procesado significativo sobre ellos) como del cliente (que puede almacenarlos en memoria caché).

Debe observarse que, aunque la FIG. 5 muestra instrucciones de archivos de plantillas estáticas abstractas, que se usan para generar XHTML, y la FIG. 8 muestra instrucciones de archivos de plantillas estáticas literales para generar RTF, esto simplemente tiene fines ilustrativos. No hay nada del XHTML que requiera instrucciones de archivos de plantillas estáticas abstractas, y no hay nada del RTF que requiera instrucciones de archivos de plantillas estáticas literales. El XHTML, el RTF y otros tipos de contenido (por ejemplo, diagramas, gráficos, informes, documentos, hojas de cálculos, páginas web, etcétera) se pueden generar usando instrucciones de archivos de plantillas estáticas abstractas y/o literales.

Por otra parte, las instrucciones de archivos de plantillas estáticas abstractas, mostradas en la FIG. 5, no se limitan a la generación de XHTML. Se podría usar el mismo archivo de plantillas estáticas para generar un informe RTF similar al mostrado en la FIG. 10, simplemente ordenando al motor de procesado del cliente que genere RTF en lugar de XHTML. De esta manera, un archivo dado de plantillas estáticas y el conjunto de datos dinámicos asociado se podrían usar para generar muchos tipos diferentes de contenido (por ejemplo, informes, documentos, hojas de cálculo, páginas web, etcétera), siempre que el motor de procesado del cliente soporte los tipos de contenido objetivo).

Tal como se ha descrito previamente, el dispositivo integrado puede delegar al cliente otras tareas además de la generación de contenidos. Por ejemplo, un archivo de plantillas estáticas puede contener instrucciones para recuperar un conjunto de datos dinámicos, transformarlo, y a continuación presentarlo de vuelta al dispositivo integrado. Esta capacidad permite que clientes funcionen como nodos en una red informática distribuida, con el dispositivo integrado transfiriendo tareas intensivas, desde el punto de vista computacional, a uno o más clientes que

proporcionen soporte de procesado distribuido para un dispositivo integrado dado en cualquier instante de tiempo dado.

La FIG. 5 y la FIG. 8 se proporcionan con fines ilustrativos, y no pretenden fijar límites o restricciones al alcance de las instrucciones de procesado de archivos de plantillas estáticas. Por ejemplo, las instrucciones de procesado de archivos de plantillas estáticas se podrían usar para implementar un sistema de acontecimientos con el fin de responder a acciones de usuario o para implementar un sistema para entradas, por parte del usuario, que validen números (por ejemplo, especificar valores permitidos mínimo, máximo, y especial) y/o cadenas de texto (por ejemplo, especificar patrones de texto permitidos). Un punto importante es que los archivos de plantillas estáticas están destinados y diseñados para funcionar como plantillas de "producción en serie" que proporcionan instrucciones para generar contenido y/o instrucciones de procesado computacional al motor de procesado del cliente, con el fin de minimizar recursos requeridos dentro del dispositivo integrado y con el fin de transferir trabajo desde el dispositivo integrado al cliente.

Conjuntos de datos dinámicos

Haciendo referencia a la FIG. 2, el segundo elemento es el conjunto de datos dinámicos 204. Cada conjunto de datos dinámicos contiene un conjunto (es decir, una "colección" o "paquete") de datos que se intercambia entre el dispositivo integrado 102 y el cliente 104. A diferencia de los archivos de plantillas estáticas, que son recursos estáticos, cada conjunto de datos dinámicos es generado dinámicamente por el dispositivo integrado (o cliente). Son formatos ejemplificativos para conjuntos de datos dinámicos el JSON y el XML. El JSON es sencillo de analizar sintácticamente y de generar, puede representar objetos de datos de complejidad arbitraria, y se diseñó específicamente para proporcionar una representación compacta de datos. La FIG. 4 es un ejemplo de un conjunto de datos dinámicos simple en formato JSON con diez elementos de datos (separado cada uno de ellos por una coma), el cual se usa para generar la página web de la FIG. 7 y el informe de la FIG. 10. Tal como puede observarse en la FIG. 4, el conjunto ejemplificativo de datos dinámicos de JSON está compuesto casi en su totalidad por datos puros.

El uso de una representación compacta y (casi) pura de datos y la minimización del procesado de datos dinámicos por parte del dispositivo integrado, tal como se ha descrito previamente, son muy importantes para optimizar (es decir, aumentar) la velocidad de transferencia de conjuntos de datos dinámicos y mejorar la escalabilidad y la sensibilidad del dispositivo integrado. Por ejemplo, un informe que muestre resultados del rendimiento de fabricación por turno durante el último año puede contener datos para 1.000 turnos, donde cada turno viene representado por una parte (por ejemplo, un registro de datos) de un conjunto grande de datos dinámicos (por ejemplo, múltiples registros de datos). Alternativamente, una página web que muestre resultados actuales de fabricación se puede actualizar múltiples veces por segundo, generando de este modo muchos miles de transacciones de conjuntos de datos dinámicos en una hora. En ambos casos, es probable que la cantidad de datos dinámicos generados sea significativa, y por lo tanto, es probable también que lo que se ahorra con una representación de datos compacta sea también significativo. El espacio ocupado en memoria de un conjunto de datos dinámicos se puede reducir adicionalmente comprimiéndolo antes de transferirlo al cliente (por ejemplo, usando una compresión gzip).

Una representación compacta de datos conserva RAM y otros recursos dentro del dispositivo integrado. El uso de recursos dentro del dispositivo integrado se puede reducir adicionalmente generando la respuesta a una solicitud de conjunto de datos dinámicos de cliente en múltiples partes, de tal manera que solamente una parte de la respuesta se almacena en la memoria del dispositivo integrado en un momento cualquiera. Por ejemplo, un conjunto de datos dinámicos que contiene diez registros de datos se puede generar en diez partes, una para cada registro, de tal manera que solamente uno de los diez registros de datos de la respuesta se almacene en la RAM del dispositivo integrado en un momento cualquiera.

La información necesaria para construir una solicitud de conjunto de datos dinámicos puede estar asociada implícitamente a un archivo de plantillas estáticas o se puede incluir explícitamente dentro de un archivo de plantillas estáticas. Los ejemplos de archivo de plantillas estáticas de la FIG. 5 y la FIG. 8 se basan ambos en una asociación implícita con el recurso del archivo de plantillas estáticas (nada del archivo de plantillas estáticas identifica explícitamente la solicitud de conjunto de datos dinámicos). La inclusión explícita de una solicitud de conjunto de datos dinámicos dentro de un archivo de plantillas estáticas puede ser tan sencilla como incluir un elemento conjunto_datos_dinámicos con un atributo de consulta tal como query="SELECT * FROM Table1". Alternativamente, el archivo de plantillas estáticas puede remitir explícitamente a uno o más datos dentro del dispositivo integrado. Por ejemplo, el atributo de índice 502 (FIG. 5) se podría modificar para remitir a un dato particular dentro del dispositivo integrado, construyendo automáticamente el motor de procesado del cliente una consulta de conjunto de datos dinámicos a partir de los atributos de índice.

Aunque no es necesario, la especificación y generación de conjuntos de datos dinámicos se puede simplificar organizando lógicamente en tablas datos dentro del dispositivo integrado (como en una base de datos). Los datos a continuación se pueden consultar usando sentencias SELECT SQL convencionales incluso si la gestión de datos subyacentes en el dispositivo integrado no es una base de datos siempre que el dispositivo integrado incluya un "intérprete" adecuado para el SQL. Esto es consistente con la interpretación del dispositivo integrado como un

servidor de datos. Además, usando la organización tabular, se puede referenciar cualquier dato en el dispositivo integrado con un simple identificador compuesto por una ID de tabla, una ID de fila, y una ID de columna.

Un único conjunto de datos dinámicos puede contener una cantidad arbitraria de datos, que pueden estar organizados linealmente (por ejemplo, como datos tabulares) o jerárquicamente (por ejemplo, como un "objeto" de complejidad arbitraria).

Los conjuntos de datos dinámicos pueden ser generados por el dispositivo integrado y transmitidos al cliente, o alternativamente pueden ser generados por el cliente y ser transmitidos al dispositivo integrado. Por ejemplo, los datos introducidos en el cliente por el usuario se pueden "enviar" al dispositivo integrado como un conjunto de datos dinámicos.

Los conjuntos de datos dinámicos también se pueden cifrar para proporcionar seguridad de datos, la cual consiga que este sistema resulte adecuado para su uso en aplicaciones de alta seguridad.

Los conjuntos de datos dinámicos pueden ser procesados por el motor de procesado del cliente para generar información nueva a partir de elementos de datos existentes (por ejemplo, calcular sumas y medias) sin necesitar un trabajo adicional del dispositivo integrado. El motor de procesado del cliente puede realizar muchos tipos diferentes de manipulación y transformación de datos, tales como filtrado, clasificación, agrupamiento y resumen. Esta técnica puede ser especialmente útil cuando el cliente acumula datos de múltiples dispositivos integrados tal como se describe posteriormente.

Debe observarse que los métodos y técnicas descritos en la presente memoria se proporcionan con fines ilustrativos, y no pretenden fijar límites o restricciones en el alcance de las operaciones de los conjuntos de datos dinámicos. Un punto importante es que los conjuntos de datos dinámicos están destinados a y diseñados para encapsular el aspecto "dinámico" del contenido con una representación compacta, con el fin de minimizar los recursos requeridos dentro del dispositivo integrado y con el fin de transferir trabajo desde el dispositivo integrado al cliente.

Motor de procesado del cliente

En referencia a la FIG. 2, el tercer elemento es el motor de procesado del cliente 206, que es un archivo de recursos estáticos, o pluralidad de archivos de recursos estáticos, almacenados en el dispositivo integrado 102 (por ejemplo, en memoria flash) y transmitidos al cliente 104 según se requiera. El motor de procesado del cliente es un archivo de recursos estáticos, lo cual significa que el dispositivo integrado simplemente lee el archivo de su memoria y lo transmite al cliente. No es necesario ningún otro procesado significativo en el dispositivo integrado.

Una vez que el mismo ha sido cargado por el cliente 104, el motor de procesado del cliente 206 es un programa ejecutable, tal como un programa JavaScript, que interpreta archivos de plantillas estáticas y lleva a cabo las instrucciones especificadas en los mismos. La ejecución de dichas instrucciones requiere generalmente que el motor de procesado del cliente disponga de capacidades adicionales. En referencia a la FIG. 3, el motor de procesado del cliente puede comprender un núcleo de procesado 302 (que dirige y controla los otros sub-componentes del motor de procesado del cliente), un procesador de archivos de plantillas estáticas 304, un generador de interfaces de usuario 306, un generador de informes 308, un formateador de datos 310, un validador de datos 312, un gestor de canales de comunicación 314, un rebotador de archivos 316 (que memoriza y carga archivos en y desde el ordenador anfitrión y/o sistemas de archivos accesibles para el ordenador anfitrión), y posiblemente componentes adicionales 318. Un punto importante es que el motor de procesado del cliente 206 está explícitamente destinado a y diseñado para minimizar recursos requeridos dentro del dispositivo integrado y para transferir trabajo desde el dispositivo integrado al cliente. De este modo, la FIG. 3 se proporciona con fines ilustrativos, y no pretende fijar límites o restricciones en el alcance del motor de procesado del cliente.

Con el estado actual de la tecnología de clientes, escribir el motor de procesado del cliente en el lenguaje JavaScript y escribir archivos de plantillas estáticas en XML puede proporcionar ventajas significativas, ya que permite que el motor de procesado del cliente se aproveche de capacidades que son nativas para los clientes, tales como la disponibilidad de funcionalidad exhaustiva para analizar sintácticamente, manipular, y trabajar de otro modo con documentos XML, lo cual puede reducir drásticamente la complejidad del motor de procesado del cliente.

El motor de procesado del cliente puede ser tan simple o complejo como requiera una aplicación en particular (es decir, tipo de dispositivo integrado). Por ejemplo, se puede construir un motor importante (aunque limitado) de procesado del cliente con 5 KB ó menos de JavaScript, mientras que uno que sea rico en características podría incluir 50 KB ó más de JavaScript.

El motor de procesado del cliente también se puede diseñar como una serie de módulos (es decir, archivos más pequeños) que se carguen dinámicamente (que se transmitan al cliente bajo demanda). Esto reduce el tiempo de carga inicial para un cliente nuevo. Igual que otros archivos de recursos estáticos, el(los) archivo(s) que comprende(n) el motor de procesado del cliente también se puede(n) almacenar de una manera comprimida (por

ejemplo, formato gzip), lo cual reduce tanto el espacio ocupado de memoria en el dispositivo integrado como el tiempo de carga inicial. El espacio ocupado por la memoria en el dispositivo integrado se puede reducir todavía más cargando el motor de procesado del cliente o módulos del mismo desde una o más fuentes que no sean el dispositivo integrado (por ejemplo, desde un anfitrión externo según remitan otros recursos cargados desde el dispositivo integrado).

El motor de procesado del cliente juega un papel central en la mejora de la escalabilidad del sistema, ya que se puede ejecutar simultáneamente en un número cualquiera de clientes. Además, el motor de procesado del cliente realiza tanto tareas de presentación como tareas lógicas comerciales (entre los ejemplos de estas últimas se incluyen gestión de comunicación, realización de validación de datos y manipulación de datos, y generación de informes complejos), lo cual eleva al cliente de ser un medio de entrega de contenido a ser un "nodo" de procesado en un sistema informático distribuido.

Canal de comunicación gestionado

En referencia a la FIG. 2, el cuarto elemento es el canal de comunicación gestionado 210. Los canales de comunicación gestionados son usados por el cliente 104, específicamente el motor de procesado del cliente 206, para comunicarse con uno o más dispositivos integrados. El término "gestionado" se refiere al hecho de que el canal de comunicación es controlado por el motor de procesado del cliente, no directamente por el cliente tal como se produce tradicionalmente. Los canales de comunicación gestionados permiten que el motor de procesado del cliente mantenga la comunicación en curso con uno o más dispositivos integrados, transfiriendo información entre el cliente y el dispositivo integrado de una manera ininterrumpida que es invisible para el usuario, de manera que se proporciona una experiencia superior para el usuario. Por ejemplo, el motor de procesado del cliente puede usar un canal de comunicación gestionado para recuperar información que se usa con el fin de actualizar dinámicamente una página web que muestra datos de rendimiento de fabricación en tiempo real (por ejemplo, actualizándose diez veces por segundo) sin parpadeos, actualizaciones de páginas u otros artefactos visuales.

Los canales de comunicación gestionados 210 también pueden ser usados por el cliente para recuperar código ejecutable (tal como JavaScript) del dispositivo integrado 102 "bajo demanda" (es decir, solo cuando realmente sea necesario), tal como cuando el motor de procesado del cliente 206 está diseñado en forma de una serie de módulos cargados dinámicamente (tal como se ha descrito previamente).

Existen varios métodos diferentes que se pueden usar para implementar un canal de comunicación gestionado. Dos métodos muy útiles son el XMLHttpRequest (al que se hace referencia también como XHR) y la manipulación de documentos.

El XHR es una norma de facto para la comunicación cliente-servidor HTTP y está disponible en varias formas para las versiones actuales de clientes populares, tales como los navegadores web Internet Explorer® 6 e Internet Explorer® 7 de Microsoft Corporation, los navegadores web Firefox® 1.5 y Firefox® 2 de Mozilla Corporation, el navegador web Safari™ 2 de Apple Inc., y el navegador web Opera™ 9 de Opera Software ASA. El XHR se puede usar para solicitar datos del dispositivo integrado (usando GET) o para enviar datos al dispositivo integrado (usando POST). Además, las solicitudes XHR se pueden designar o bien como síncronas o bien como asíncronas según se necesite. No obstante, el XHR se limita en general al acceso a recursos en el dominio local.

El XHR también se puede usar indirectamente para cargar código ejecutable bajo demanda recuperando ese código en forma de una cadena de texto. Por ejemplo, lo siguiente ilustra una técnica JavaScript conocida de plataforma cruzada que consigue que el código de una cadena de texto forme parte del ámbito global (en donde es accesible para el motor de procesado del cliente):

```
función añadir_a_ámbito_global (cadena_código)
{
        var global = esto;
        si (window.execScript)
        {
                window.execScript (cadena_código);
                return null;
        }
        return global.eval ? global.eval (cadena_código):
eval (cadena_código);
}
```

Por otro lado, la manipulación de documentos resulta particularmente útil si se requiere un acceso a dominio no local (es decir, cruzado). La manipulación de documentos es una técnica menos conocida pero muy útil para la comunicación gestionada, en la que se descarga un recurso añadiendo dinámicamente un identificador nuevo a una

página web existente, en donde dicho identificador remite al recurso deseado. Por ejemplo, se puede descargar un archivo JavaScript añadiendo dinámicamente un identificador <script> a la página actual, tal como <script src="http://192.168.1.240/data_1138.js">. El DOM del W3C (Modelo de Objeto de Documento del Consorcio de la Malla Multimedia Mundial), el método document.write() de JavaScript, y la propiedad innerHTML se pueden usar todos ellos para realizar la manipulación de documentos.

En referencia a la FIG. 1, un uso importante de la manipulación de documentos en esta forma de realización es solicitar conjuntos de datos dinámicos de múltiples dispositivos integrados 102 (es decir, de múltiples dominios y/o direcciones IP). Esto permite que cualquier cliente individual 104 con un motor de procesado del cliente 206 (FIG. 2) solicite, manipule, transforme, acumule, y presente datos de una pluralidad de dispositivos integrados. Por ejemplo, el dispositivo integrado puede servir un conjunto de datos dinámicos en forma de un archivo JavaScript generado dinámicamente dando formato a los datos como JSON y asignando esos datos formateados a una variable JavaScript (consiguiendo de este modo que los datos formen parte de una sentencia JavaScript completa). Cada vez que se remita a dicho archivo en el cliente (por ejemplo, a través de un identificador <script> generado dinámicamente tal como se ha descrito anteriormente), el mismo será solicitado automáticamente desde el dispositivo integrado que se especifica en el URL del recurso (es decir, en el atributo src del identificador <script>).

Los canales de comunicación gestionados están vinculados solo de manera moderada con los métodos de comunicación antes descritos. Los mismos se pueden adaptar fácilmente para usar otros métodos a medida que evolucionen las normas (de facto u otras) de Internet, tales como la norma W3C propuesta "Document Object Model (DOM) Level 3 Load and Save Specification".

Algunas de las técnicas antes descritas funcionarán con una comunicación tanto síncrona como asíncrona. Se recomienda la comunicación asíncrona ya que en general proporciona una experiencia superior para el usuario. Debe observarse que esto constituye una preferencia, no un requisito. En muchos casos, el motor de procesado del cliente seguirá funcionando de manera eficaz si los canales de comunicación gestionados usan una comunicación síncrona en lugar de una comunicación asíncrona.

Debe observarse que los métodos y técnicas descritos en la presente memoria se proporcionan con fines ilustrativos, y no pretenden fijar límites o restricciones en el alcance de las operaciones de los canales de comunicación gestionados. Un punto importante es que los canales de comunicación gestionados actúan como enlaces de comunicación bidireccionales entre el motor de procesado del cliente que se ejecuta en un cliente y uno o más dispositivos integrados. Los mismos se usan para intercambiar información que soporta los objetivos de minimizar los recursos requeridos dentro del dispositivo integrado y transferir trabajo desde el dispositivo integrado al cliente.

Interacción de elementos principales

Para entender mejor cómo interaccionan el motor de procesado del cliente, los archivos de plantillas estáticas, los conjuntos de datos dinámicos, y los canales de comunicación gestionados, se muestran las etapas necesarias para generar una página web con elementos tanto estáticos como dinámicos, en forma de un diagrama de secuencias en la FIG. 11, y las mismas se describen de forma detallada a continuación.

En una forma de realización de la invención, según se ilustra en la FIG. 11, se genera contenido puramente estático de la manera siguiente:

1. El usuario solicita una página web específica (por ejemplo, siguiendo un enlace o introduciendo directamente un URL) (etapa 1102).

2. El cliente se conecta el dispositivo integrado y solicita la página web específica (URL) (etapa 1104).

3. El dispositivo integrado transmite la página web solicitada al cliente. A esta página se le hace referencia como página de "arranque", puesto que en lugar del contenido solicitado por el usuario, contiene los medios para generar este contenido, a través de referencias al motor de procesado del cliente, uno o más archivos de plantillas estáticas, y opcionalmente uno o más conjuntos de datos dinámicos y otros recursos (etapa 1106).

4. La página de arranque contiene una referencia al motor de procesado del cliente (por ejemplo, a través de un identificador <script> HTML), que provoca que el cliente solicite el motor del procesado del cliente del dispositivo integrado (etapa 1108).

5. El dispositivo integrado transmite el motor de procesado del cliente al cliente (etapa 1110).

6. La página de arranque recibida en el punto 3 anterior contiene una llamada a la función de entrada en el motor de procesado del cliente, que provoca que el cliente llame a la función de entrada y comience a ejecutar el motor de procesado del cliente (etapa 1112).

7. La página de arranque recibida en el anterior punto 3 especifica el URL de un archivo de plantillas estáticas que contiene instrucciones para generar el contenido deseado, y el motor de procesado del cliente solicita ese archivo de plantillas estáticas usando un canal de comunicación gestionado (etapa 1114).

8. El dispositivo integrado transmite el archivo de plantillas estáticas al cliente (etapa 1116).

9. El motor de procesado del cliente analiza sintácticamente el archivo de plantillas estáticas y genera el contenido estático asociado de manera correspondiente a partir de instrucciones de procesado abstractas y/o literales (etapa 1118).

10. El contenido estático está preparado (etapa 1120).

Cuando el contenido incluye elementos dinámicos, se realizan las siguientes etapas adicionales:

11. El motor de procesado del cliente identifica un requisito de conjunto de datos dinámicos asociado al archivo de plantillas estáticas (por ejemplo, una sentencia SELECT SQL que está incluida en el archivo de plantillas estáticas). El requisito de conjunto de datos dinámicos puede ser implícito o explícito tal como se ha descrito anteriormente (etapa 1122).

12. El motor de procesado del cliente usa un canal de comunicación gestionado para solicitar un conjunto de datos dinámicos (etapa 1124).

13. El dispositivo integrado genera el conjunto de datos dinámicos y lo transmite al cliente (etapa 1126).

14. El motor de procesado del cliente transforma los datos del conjunto de datos dinámicos basándose en instrucciones de procesado del archivo de plantillas estáticas y acumula los datos transformados y el contenido estático del punto 10 (etapa 1128); y

15. Para contenido que cambia con el tiempo, tal como una página de rendimiento de fabricación en tiempo real, el motor de procesado del cliente repite los puntos 12 a 14 siempre que la página web se mantenga abierta (etapa 1130).

Las etapas anteriores se centran en las interacciones entre el archivo de plantillas estáticas, el conjunto de datos dinámicos, el motor de procesado del cliente y el canal de comunicación gestionado. En implementaciones reales, es probable que se carguen archivos de recursos estáticos adicionales, tales como archivos CSS (Hojas de Estilo en Cascada), archivos de imágenes, etcétera, o bien directamente, tal como a través de referencias en la página web de arranque, o bien indirectamente como solicitudes de recursos del motor de procesado del cliente a través de un canal de comunicación gestionado. Debe observarse también que una página de arranque puede remitir a más de un archivo de plantillas estáticas, en cuyo caso los puntos 7 a 9 se repetirán para cada archivo de plantillas estáticas. De manera similar, puede haber más de un conjunto de datos dinámicos asociado a un archivo de plantillas estáticas dado, en cuyo caso se repetirán los puntos 11 a 13 hasta que se hayan recibido todos los conjuntos de datos dinámicos. Resultará evidente para los expertos en la materia, que, en las etapas antes descritas, se pueden realizar estas y otras modificaciones sin desviarse con respecto al espíritu y el alcance de esta forma de realización particular.

Haciendo referencia a la FIG. 2, los archivos de recursos estáticos, tales como los archivos de plantillas estáticas 208, el motor de procesado del cliente 206, y otros según se ha mencionado anteriormente, deberían ser almacenados en memoria caché por el cliente 104 siempre que sea posible. El cliente lleva a cabo esto almacenando copias de los archivos de plantillas estáticas, el motor de procesado del cliente, y otros archivos en el ordenador anfitrión 106. Puesto que el recurso es estático y, por lo tanto, no varía, el cliente puede satisfacer solicitudes adicionales para el mismo recurso usando esta copia almacenada, ahorrando tiempo y recursos dentro del dispositivo integrado 102 (que no necesita retransmitir el mismo archivo). El almacenamiento en memoria caché del cliente puede reducir significativamente la carga de trabajo del dispositivo integrado, así como mejorar la sensibilidad del cliente, de manera que es importante que el dispositivo integrado proporcione soporte para el almacenamiento en memoria caché del cliente (por ejemplo, usando encabezamientos HTTP apropiados). Haciendo referencia a la FIG. 11, debe observarse que si la página de arranque transmitida por el dispositivo integrado 102 en la etapa 1106, el motor de procesado del cliente transmitido por el dispositivo integrado en la etapa 1110, y el archivo de plantillas estáticas transmitido por el dispositivo integrado en la etapa 1116 son almacenados en memoria caché por el cliente (tal como debería producirse típicamente), el dispositivo integrado no necesitará transmitir los mismos nuevamente. Debe observarse también que, con el almacenamiento en memoria caché del cliente, el motor de procesado del cliente se recibe una vez y, a continuación, está disponible para cada una de las páginas de arranque cargadas desde el dispositivo integrado.

Haciendo referencia nuevamente a la FIG. 2, debe observarse que el dispositivo integrado 102 no tiene que realizar ningún procesado significativo en relación con su archivo de plantillas estáticas almacenado 208 y los archivos del motor de procesado del cliente 206. Simplemente los transmite al cliente 104. De manera similar, cada conjunto de

datos dinámicos 204 requiere un procesado mínimo del dispositivo integrado. Los conjuntos de datos dinámicos son datos (casi) puros con un formateo mínimo; y el motor de procesado del cliente puede leer instrucciones del archivo de plantillas estáticas y transformar los datos de manera correspondiente. Tal como se ha descrito anteriormente, el dispositivo integrado actúa como un servidor de archivos y datos, y el motor de procesado del cliente es responsable de transformar archivos de plantillas estáticas y conjuntos de datos dinámicos en contenido útil. Esta división del trabajo transfiere una gran cantidad de tarea de procesado desde el dispositivo integrado a los clientes de red, mejorando significativamente la escalabilidad de la solución global.

Rebote de archivos

Otra característica de varias formas de realización de la presente invención es la capacidad de leer archivos arbitrariamente grandes del ordenador anfitrión (incluyendo la lectura de dichos archivos de cualesquiera sistemas de archivos accesibles para el ordenador anfitrión) y de escribir archivos arbitrariamente grandes en el ordenador anfitrión (incluyendo la escritura o el almacenamiento de dichos archivos en cualesquiera sistemas de archivos accesibles para el ordenador anfitrión y la abertura de dichos archivos en el ordenador anfitrión). Por ejemplo, el usuario puede desear almacenar un informe generado por el cliente (por ejemplo, un informe generado por el motor de procesado del cliente) en el ordenador anfitrión para acceder al mismo posteriormente o para archivarlo. Como ejemplo adicional, un dispositivo integrado puede necesitar acceder a un archivo que está almacenado en el ordenador anfitrión con el fin de realizar ciertas tareas, pero dispone de limitaciones de recursos que le evitan almacenar el archivo dentro de su memoria.

La implementación de esta característica puede resultar problemática ya que los clientes (por ejemplo, navegadores web) pueden controlar minuciosamente el acceso a sistemas de archivos del ordenador anfitrión por razones de seguridad. La finalidad principal de este control es típicamente evitar que programas potencialmente maliciosos (por ejemplo, JavaScript malicioso) accedan al sistema de archivos. Estas restricciones hacen también que resulte difícil, cuando no imposible, que el motor de procesado del cliente lea y escriba archivos desde y en el ordenador anfitrión por sí mismo, incluso con el permiso explícito del usuario. Los clientes de navegadores web sí permiten típicamente que el usuario conceda permiso para cargar archivos accesibles para el ordenador anfitrión en un servidor HTTP y descargar archivos desde un servidor HTTP al ordenador anfitrión (debería observarse que, en dichas operaciones, podrían participar dispositivos integrados que incluyen servidores HTTP). No obstante, tal como se ha mencionado anteriormente, el dispositivo integrado puede que no disponga de los recursos necesarios para almacenar un archivo completo, o ni siquiera una parte significativa de un archivo, en memoria, ni siquiera durante un periodo de tiempo breve. Esto es así particularmente cuando es deseable que el dispositivo integrado pueda gestionar múltiples archivos de esta manera simultáneamente. Además, tal como se ha mencionado previamente, es necesario un mecanismo para que el contenido (por ejemplo, informes, documentos, hojas de cálculo, etcétera) generado por el motor de procesado del cliente pueda ser almacenado o abierto por el ordenador anfitrión.

La solución a estos problemas es un sistema y un método que afrontan las restricciones de seguridad antes mencionadas, aunque, al mismo tiempo, permitiendo que dispositivos integrados de recursos limitados lean y escriban archivos arbitrariamente grandes desde y en el ordenador anfitrión (con el permiso del usuario).

El rebote de archivos facilita los siguientes escenarios:

- Los archivos del ordenador anfitrión, con el permiso del usuario, se pueden cargar en el cliente, que, con frecuencia, dispondrá de una memoria órdenes de magnitud mayor que el dispositivo integrado. Una vez cargados dentro del cliente, se puede acceder a los archivos y los mismos se pueden manipular a deseo por medio del dispositivo integrado, usando, como su intermediario, el motor de procesado del cliente.

- El contenido generado por el cliente (típicamente por medio del funcionamiento del motor de procesado del cliente como proxy para el dispositivo integrado), con el permiso del usuario, se puede almacenar o abrir como un archivo en el ordenador anfitrión.

Con el rebote de archivos, el cliente envía el archivo (es decir, el contenido del archivo) al dispositivo integrado en una serie de uno o más paquetes, que el dispositivo integrado simplemente “hace rebotar” de vuelta al cliente (de aquí el nombre “rebote de archivos”). El dispositivo integrado conserva únicamente un paquete dado durante el tiempo que se tarda en hacer rebotar ese paquete de vuelta al cliente, después de lo cual el paquete se descarta. De este modo, el dispositivo integrado no necesita almacenar el archivo completo, y realiza un procesado mínimo sobre los paquetes. Los paquetes pueden tener un tamaño arbitrario, y el dispositivo integrado únicamente necesita reservar recursos de memoria suficientes para almacenar en memoria intermedia un único paquete cada vez (aunque se puede mejorar el rendimiento almacenando en memoria intermedia simultáneamente múltiples paquetes). Cuando se “hace rebotar” simultáneamente más de un archivo, se dice que cada archivo usa un “canal” de rebote de archivos diferente.

El rebote de archivos se podría realizar haciendo que el cliente envíe un paquete (es decir, una parte del archivo) al dispositivo integrado, de manera que el dispositivo integrado envía el paquete de vuelta al cliente y espera a que el cliente envíe el siguiente paquete, y el proceso se repite hasta que no haya más paquetes a enviar (es decir, se ha

"hecho rebotar" el archivo completo). No obstante, no todos los protocolos y/o clientes de comunicación soportan este tipo de comportamiento. Por ejemplo, un protocolo de comunicación según se especifica o según se implementa (tal como el HTTP) podría requerir que el cliente enviase su solicitud completa (es decir, el archivo completo) antes de que el servidor (es decir, el dispositivo integrado) pueda comenzar a enviar su respuesta. Por lo tanto, una solución más generalizada es usar dos conexiones cooperativas para lograr un resultado similar: los paquetes recibidos desde el cliente a través de la primera conexión se devuelven al cliente a través de la segunda conexión, tal como se describe posteriormente y según se ilustra en la FIG. 12 y la FIG. 13.

En una forma de realización de la invención, las etapas para almacenar (es decir, escribir) un archivo generado por el cliente en el ordenador anfitrión se muestran en la FIG. 12, y se describen en líneas generales de forma más detallada posteriormente. Resultará evidente para los expertos en la materia que, en las etapas descritas en el presente caso, se pueden aplicar varias modificaciones sin desviarse con respecto al espíritu y el alcance de esta forma de realización.

1. El usuario realiza una solicitud que invocará el rebote de archivos de alguna manera específica de la aplicación (etapa 1202).

2. El motor de procesado del cliente usa un canal de comunicación gestionado para solicitar una ID de canal de rebote de archivos (cada ID de canal reserva recursos en el dispositivo integrado, incluyendo los recursos para dos conexiones HTTP) (etapa 1204).

3. El dispositivo integrado busca una ID de canal disponible (etapa 1206).

   - Si tiene una ID de canal disponible, marca el canal correspondiente como reservado.
   - Si no tiene una ID de canal disponible, en este momento no se puede realizar un rebote de archivo (se envía una respuesta apropiada al cliente, el cual, a su vez, visualiza un mensaje para el usuario, y el proceso finaliza aquí).

4. El dispositivo integrado transmite la ID de canal correspondiente al canal reservado en el punto anterior 3 al cliente (etapa 1208).

5. Si todavía no se ha generado el archivo a almacenar (por ejemplo, un informe), el motor de procesado del cliente realiza cualquier acción necesaria para generar el archivo (etapa 1210).

6. El motor de procesado del cliente usa un canal de comunicación gestionado para iniciar un POST del archivo generado, junto con la ID de canal y cualesquiera otros metadatos asociados (por ejemplo, nombre de archivo, tamaño de archivo, tipo MIME de archivo, etcétera) hacia el dispositivo integrado. Debe observarse que el dispositivo integrado coloca los datos de archivos (es decir, contenido, no metadatos) incluidos en este POST en una memoria intermedia asociada a la ID de canal especificada (etapa 1212).

7. El motor de procesado del cliente al mismo tiempo emite una solicitud GET que incluye la ID de canal como parámetro de URL para iniciar la recuperación del archivo que se está haciendo rebotar. Esta solicitud se realiza a través de un marco flotante (*iframe*) oculto en la página web para separar claramente la solicitud GET de la solicitud POST. Algunos clientes pueden plantear restricciones sobre la abertura del marco flotante por acción del programa, en cuyo caso, para activar su creación se puede usar una acción del usuario tal como un clic de un botón (etapa 1214).

8. El dispositivo integrado transmite los encabezamientos HTTP apropiados en respuesta a la solicitud GET del punto 7 anterior, incluyendo un encabezamiento Disposición de Contenido HTTP para el archivo que está haciendo rebotar. La primera parte del archivo también puede ser transmitida por el dispositivo integrado en este momento. Consúltese el uso de la memoria intermedia según se describe en los puntos 10 y 11 posteriores (etapa 1216).

9. Cuando el cliente (navegador) recibe el encabezamiento de Disposición de Contenido, abre un recuadro de diálogo que permite que el usuario seleccione si el archivo se debería abrir o memorizar en el ordenador anfitrión cuando se haya completado la descarga (etapa 1218).

10. Como consecuencia de la solicitud de POST iniciada en el punto 6 anterior, el dispositivo integrado recibe del cliente datos que equivalen hasta una ventana de TCP, los cuales se sitúan en una memoria intermedia asociada a la ID de canal especificada (etapa 1220).

11. El dispositivo integrado extrae datos (es decir, contenido de archivo) de la memoria intermedia asociada a la ID de canal especificada (véase el punto 10 anterior) y los transmite al cliente (a través de la respuesta GET iniciada en el punto 7 anterior) (etapa 1222).

12. Los puntos 10 y 11 anteriores se repiten hasta que se ha “hecho rebotar” el archivo completo, momento en el que el archivo será abierto o memorizado en el ordenador anfitrión (dependiendo de la selección del usuario en el punto 9 anterior) (etapa 1224).

13. El dispositivo integrado marca la ID de canal como libre (etapa 1226).

14. El dispositivo integrado envía una respuesta apropiada (específica de la aplicación) al cliente para confirmar que se ha completado la solicitud POST del punto 6 anterior (etapa 1228).

En una forma de realización de la invención, las etapas para cargar (es decir, leer) un archivo desde el ordenador anfitrión al cliente (navegador web) se muestran en la FIG. 13, y se describen en líneas generales de forma más detallada posteriormente. Resultará evidente para los expertos en la materia que, en las etapas descritas en el presente caso, se pueden aplicar varias modificaciones sin desviarse con respecto al espíritu y el alcance de esta forma de realización.

1. El usuario navega a una página web que invocará el rebote de archivos para seleccionar (y finalmente cargar) un archivo desde el ordenador anfitrión. A esta página se le hace referencia como madre (etapa 1302).

2. El cliente se conecta al dispositivo integrado y solicita la página web madre (etapa 1304).

3. El dispositivo integrado transmite la madre al cliente, que contiene un marco flotante (*iframe*) (al que se hace referencia como hijo). Debe observarse que el hijo se usa para seleccionar el archivo y aplicarle un POST y la madre se usa para aplicar al archivo un GET y acceder al mismo (etapa 1306).

4. El cliente solicita el contenido del cuadro flotante (es decir, el hijo) del dispositivo integrado (etapa 1308).

5. El dispositivo integrado transmite el hijo (que incluye un control de carga de archivos basado en formularios HTML que se usará para seleccionar el archivo a cargar) hacia el cliente (etapa 1310).

6. El motor de procesado del cliente usa un canal de comunicación gestionado (desde el hijo) para solicitar una ID de canal de rebote de archivos desde el dispositivo integrado (cada ID de canal reserva recursos en el dispositivo integrado, incluyendo los recursos para dos conexiones HTTP) (etapa 1312).

7. El dispositivo integrado busca una ID de canal disponible (etapa 1314).

   - Si tiene una ID de canal disponible, marca el canal correspondiente como reservado.

   - Si no tiene una ID de canal disponible, no se puede realizar un rebote de archivo en este momento (se envía una respuesta apropiada al hijo, el cual, a su vez, visualiza un mensaje para el usuario, y el proceso finaliza aquí).

8. El dispositivo integrado transmite la ID de canal correspondiente al canal reservado en el punto 7 anterior al cliente (etapa 1316).

9. El hijo almacena la ID de canal en un campo oculto dentro del formulario HTML que contiene el control de carga de archivo (lo cual provoca que se cargue con el resto del formulario posteriormente) (etapa 1318).

10. El usuario selecciona el archivo deseado del ordenador anfitrión con el control de carga de archivos, y hace clic en un botón de confirmación cuando el archivo ha sido seleccionado (1320).

11. El hijo notifica a la madre que el archivo está preparado para hacerlo rebotar y proporcionar la ID de canal a la madre (etapa 1322).

12. Tras haber recibido una notificación por parte del hijo, la madre usa un canal de comunicación gestionado para iniciar una solicitud GET con el fin de recuperar el archivo desde el dispositivo integrado. El dispositivo integrado coloca esta solicitud en espera y aguarda a los datos de archivo del hijo (etapa 1324).

13. El hijo usa una solicitud POST convencional (es decir, no se requiere un canal de comunicación gestionado) para iniciar la presentación del formulario (debe observarse que, debido al control de carga de archivos basado en formularios HTML, la presentación del formulario incluye el archivo). Debe observarse que el dispositivo integrado coloca los datos del archivo (es decir, contenido, no metadatos) incluidos en este POST en una memoria intermedia asociada a la ID de canal especificada (etapa 1326).

14. El dispositivo integrado transmite los encabezamientos HTTP apropiados hacia la madre en respuesta a la solicitud GET del punto 12 anterior. En este momento el dispositivo integrado también puede transmitir la

primera parte del archivo. Consúltese el uso de la memoria intermedia según se describe en los puntos 15 y 16 a continuación (etapa 1328).

15. Como consecuencia de la solicitud POST iniciada en el punto 13 anterior, el dispositivo integrado recibe datos equivalentes hasta a una ventana de TCP, desde el hijo, y los coloca en una memoria intermedia asociada a la ID de canal especificada (etapa 1330).

16. El dispositivo integrado extrae datos (es decir, contenido del archivo) de la memoria intermedia asociada a la ID de canal especificada (véase el punto 15 anterior) y los transmite a la madre (etapa 1332).

17. Los puntos 15 y 16 anteriores se repiten hasta que se haya "hecho rebotar" el archivo completo, momento en el cual el archivo completo ha sido cargado en la madre, donde el motor de procesado del cliente puede acceder a y manipular el mismo libremente (etapa 1334).

18. El dispositivo integrado marca la ID de canal como libre (etapa 1336).

19. El dispositivo integrado envía una respuesta apropiada (específica de aplicación) al hijo para confirmar que se ha completado la solicitud POST del punto 13 anterior (etapa 1338).

Debe observarse que en ambos ejemplos descritos anteriormente en líneas generales, únicamente es necesario que resida en el dispositivo integrado una parte muy pequeña del archivo en cualquier momento. Por ejemplo, usando el rebote de archivos, se puede leer o escribir un archivo de 10 MB (o mayor) con solo una memoria intermedia RAM de 10 KB (o menor) en el dispositivo integrado, ofreciendo una reducción de 1.000 veces (o mayor) de la memoria requerida para el dispositivo integrado.

El rebote de archivos puede usar la ventana de TCP (que controla el número de bytes de datos que se pueden transmitir a través de una conexión dada sin un acuse de recibo desde el receptor) para proporcionar un control de flujo automático y un grado de sincronización entre dos conexiones cooperativas usadas para implementar un canal de rebote de archivos. A medida que se cargan datos en el dispositivo integrado, se llena la ventana TCP de la conexión POST del dispositivo integrado, y a medida que se descargan datos en el cliente a través de la conexión GET, se vacía la ventana TCP de la conexión POST del dispositivo integrado. De esta manera, la ventana TCP de la conexión POST se puede usar para "marcar el ritmo" del rebote del archivo. Esto permite una vinculación moderada entre las dos conexiones HTTP asociadas a un canal de rebote de archivos en el dispositivo integrado. El cliente únicamente transmitirá datos al dispositivo integrado (a través de la conexión POST) si el dispositivo integrado tiene espacio de memoria intermedia dentro de su ventana TCP de la conexión POST, y el dispositivo integrado simplemente transmitirá datos al cliente (a través de la conexión GET) todo lo rápido que pueda (según lo permita la ventana TCP análoga del cliente).

Este uso de dos conexiones HTTP cooperativas hace que el rebote de archivos resulte algo inusual. Por ejemplo, los servidores HTTP normalmente gestionan cada conexión de manera independiente; mientras que el rebote de archivos utiliza un grado de coordinación y cooperación entre las dos conexiones HTTP que participan en un canal dado de rebote de archivos (esencialmente están "emparejadas"). Debe observarse que, debido a que las dos conexiones HTTP usadas en el rebote de archivos funcionan simultáneamente, los canales de comunicación gestionados descritos en esta sección usan una comunicación asíncrona.

Otro aspecto inusual del rebote de archivos es cómo trata el contenido (los archivos). Normalmente, los archivos se almacenan en su totalidad dentro del servidor HTTP. Incluso los archivos generados dinámicamente se pueden considerar en general que tienen una existencia continua, en el sentido de que el servidor puede generar una copia del archivo según se necesite. No obstante, con el rebote de archivos, los archivos rebotados son completamente transitorios desde la perspectiva del servidor (es decir, del dispositivo integrado). Una vez que se ha transmitido un paquete dado desde el dispositivo integrado (y el cliente ha acusado su recibo), el mismo desaparece desde la perspectiva del dispositivo integrado (ya no es necesario).

Los ejemplos dados usan pares de conexiones HTTP; no obstante, es también posible usar conexiones FTP para el rebote de archivos. Los expertos en la materia percibirán que son posibles varias combinaciones de tipos de conexión.

Es interesante echar una mirada a resultados basados en el tiempo registrados a partir de una implementación de prueba de generación de contenido del lado del cliente combinada con rebote de archivos. Un ordenador anfitrión (con un procesador Pentium® D de 3 GHz de Intel Corporation) y un dispositivo integrado (con un núcleo ARM920T® de 200 MHz de ARM Limited) se conectaron a través de una red de área local (Ethernet de 100 Mbps). El cliente (navegador web Internet Explorer® 6 de Microsoft Corporation) recuperó una página de arranque, un motor de procesado del cliente, un archivo de plantillas estáticas (para generar un informe de Formato de Texto Enriquecido) y un conjunto de datos dinámicos del dispositivo integrado, usando canales de comunicación gestionados, según resultó apropiado, en aproximadamente 4 segundos, después de lo cual el cliente generó un informe de 1.000 páginas (3,6 MB) en aproximadamente 35 segundos, seguido por la participación del cliente y del dispositivo

integrado en una sesión de rebote de archivos para memorizar el informe en el sistema de archivos del ordenador anfitrión en aproximadamente 8 segundos.

Una de las ventajas del sistema descrito es la escalabilidad mejorada. Por ejemplo:

- El motor de procesado del cliente se puede carga en un número cualquiera de clientes, convirtiéndose cada uno de ellos en otra entidad de procesado en el sistema. Por ejemplo, un dispositivo integrado puede tener 100 clientes, cada uno de los cuales usa el motor de procesado del cliente para transferir trabajo desde el dispositivo integrado.

- Se puede usar un archivo de plantillas estáticas para generar contenido que se actualiza repetidamente (es decir, elementos dinámicos del contenido se renuevan repetidamente) en periodos de tiempo arbitrarios. Por ejemplo, el motor de procesado del cliente puede usar un archivo de plantillas estáticas para generar una página web que se renueva en tiempo real (por ejemplo, diez veces por segundo) con datos nuevos del dispositivo integrado, al mismo tiempo que se impone una carga mínima sobre el dispositivo integrado. Usando el ejemplo de la FIG. 4 (un conjunto de datos dinámicos de aproximadamente 74 caracteres), la FIG. 5 (un archivo de plantillas estáticas de aproximadamente 817 caracteres), y la FIG. 6 (un documento generado de aproximadamente 2.093 caracteres), y suponiendo que la página web se renueva diez veces por segundo durante un minuto (600 actualizaciones), se puede realizar una comparación entre el motor de procesado del cliente que genera contenido y que requiere aproximadamente 45.000 bytes para ser transmitidos desde el dispositivo integrado al cliente (817 bytes para el archivo de plantillas estáticas y 74 bytes por cada conjunto de datos dinámicos multiplicado por 600 actualizaciones) con el dispositivo integrado que genera el mismo contenido y que requiere la transmisión aproximadamente de 1.256.000 bytes desde el dispositivo integrado al cliente (2.093 caracteres por página multiplicado por 600 actualización).

- Se puede usar un archivo de plantillas estáticas para generar contenido de “profundidad” arbitraria, que actúa esencialmente como un “molde de producción en serie” de contenido. Por ejemplo, el motor de procesado del cliente puede usar un archivo de plantillas estáticas para generar cien o incluso mil informes de páginas, donde cada página se genera a partir del archivo de plantillas estáticas y un registro de un conjunto de datos dinámicos de múltiples registro.

- La división de recursos estáticos (por ejemplo, el motor de procesado del cliente y los archivos de plantillas estáticas) con respecto a recursos dinámicos (los conjuntos de datos dinámicos), maximiza la cantidad de información almacenable en memoria caché. En general, los únicos recursos que no se pueden almacenar en memoria caché son conjuntos de datos dinámicos, que están compuestos en su totalidad por datos dinámicos – la totalidad del resto de recursos puede ser almacenada típicamente en memoria caché por el cliente. Esto proporciona una ventaja muy significativa con respecto a páginas web dinámicas generadas por el lado del servidor (es decir, por el dispositivo integrado), que no se pueden almacenar en memoria caché pero que todavía podrían contener una gran cantidad de contenido estático. Esto significa también que el dispositivo integrado consume la mayor parte de su tiempo en la tarea central de servir datos dinámicos (a través de conjuntos de datos dinámicos). Esta ventaja se ve amplificada adicionalmente por el uso de datos (casi) puros en el conjunto de datos dinámicos según se ejemplifica en la forma de realización preferida.

La división entre recursos estáticos y recursos dinámicos proporciona además otras ventajas. Por ejemplo, separa claramente la información de presentación y de estilo (que se halla dentro del archivo de plantillas estáticas) con respecto a los datos (que se hallan dentro del conjunto de datos dinámicos). Esto permite que los dos varíen de manera independiente – los datos de varios conjuntos de datos dinámicos se pueden presentar con el mismo estilo, y un único conjunto de datos dinámicos se puede presentar con varios estilos diferentes. Puede verse un ejemplo de esto último en una comparación entre la FIG. 6 y la FIG. 9, que ilustra cómo se puede usar el mismo conjunto de datos dinámicos (mostrado en la FIG. 4), en combinación con dos archivos diferentes de plantillas estáticas (FIG. 5 y FIG. 8), para generar una salida notablemente diferente (la página web de la FIG. 7 y el informe RTF de la FIG. 10); o, alternativamente, cómo se puede usar el mismo conjunto de datos dinámicos (FIG. 4) y archivo de plantillas estáticas (FIG. 5) para generar una salida notablemente diferente (la página web de la FIG. 7 y el informe RTF de la FIG. 10) meramente ordenando al motor de procesado del cliente que genere un tipo diferente de contenido.

Además, esta división entre recursos estáticos y recursos dinámicos puede reducir considerablemente el procesado y los recursos requeridos para el cifrado de datos. La seguridad de los datos se puede proporcionar simplemente cifrando los conjuntos de datos dinámicos. Los recursos estáticos típicamente no contienen información privada y se pueden transferir libremente sin cifrado. Por lo tanto, el sistema descrito puede reducir considerablemente la cantidad de información a cifrar. Una vez más, esta ventaja se ve amplificada adicionalmente por el uso de datos (casi) puros en el conjunto de datos dinámicos según se ejemplifica en la forma de realización preferida.

El sistema descrito mantiene también muy bajos los requisitos de RAM del dispositivo integrado. Por ejemplo, los archivos de recursos estáticos, tales como el motor de procesado del cliente, y los archivos de plantillas estáticas se pueden almacenar en memoria flash u otros medios de almacenamiento de bajo coste, limitándose el uso de la RAM a un número pequeño de “paquetes” excepcionales cuando se está transmitiendo un archivo a un cliente. Además,

la separación de recursos estáticos y recursos dinámicos minimiza la cantidad de RAM requerida para construir y almacenar un conjunto de datos dinámicos mientras el mismo está siendo transmitido a un cliente, puesto que los conjuntos de datos dinámicos son en su totalidad datos dinámicos en lugar de una mezcla de contenido estático y dinámico. Además, el uso de rebote de archivos minimiza la memoria (típicamente RAM) requerida por el dispositivo integrado para leer y escribir archivos desde y en el ordenador anfitrión.

Aunque el sistema descrito requiere solamente recursos mínimos del dispositivo integrado, la transferencia de la generación de contenido y otro procesado al ordenador anfitrión significa que ello no va en detrimento de la experiencia del usuario. Todo lo contrario, debido a que los ordenadores anfitriones están en general órdenes de magnitud por encima de los dispositivos integrados en términos de memoria y poder de procesado, son capaces de generar contenido que es mucho mayor y más sofisticado que el que sería posible para el dispositivo integrado por sí solo.

Debería indicarse explícitamente que las formas de realización descritas no dependen de empresas, productos, o herramientas de desarrollo específicos, de terceros; y no requieren la instalación de ningún software personalizado, incluyendo ningún módulo plug-in de navegador especial, en el ordenador anfitrión. Son “genéricas” en cuanto a sus requisitos - la totalidad de los archivos requeridos (por ejemplo, el motor de procesado del cliente y el archivo de plantillas estáticas) se puede crear con un simple editor de texto (tal como el editor de texto Bloc de Notas de Microsoft Corporation) y no se imponen requisitos “especiales” sobre el ordenador anfitrión. Por ejemplo, los elementos principales de las formas de realización descritas se han sometido satisfactoriamente a prueba pasando por una gama de clientes HTTP populares, incluyendo los navegadores web Internet Explorer® 6 e Internet Explorer® 7 de Microsoft Corporation, los navegadores web Firefox® 1.5 y Firefox® 2 de Mozilla Corporation, el navegador web Safari™ 2 de Apple Inc., y el navegador web Opera™ 9 de Opera Software ASA.

Además, debería observarse que las formas de realización descritas requieren solamente un soporte HTTP muy básico del dispositivo integrado, lo cual es una consideración importante para dispositivos integrados de recursos limitados. Nada de las formas de realización descritas en la presente memoria requiere soporte para tecnologías adicionales del lado del servidor tales como PHP, CGI, o ASP.NET. Esto ayuda a mantener bajos los requisitos de recursos, permitiendo el uso de varias formas de realización de la invención incluso en sistemas integrados muy “pequeños”.

REIVINDICACIONES

1. Método para permitir que un dispositivo integrado (102) funcione conjuntamente con un cliente (104) en un ordenador anfitrión (106), para acceder a contenido de un archivo accesible para el ordenador anfitrión (106), aunque no directamente accesible para el cliente (104), que comprende las etapas siguientes:

seleccionar un archivo en el cliente (104);

establecer un enlace de comunicaciones entre el cliente (104) y el dispositivo integrado (106);

enviar una parte del archivo desde el cliente al dispositivo integrado de un tamaño que encaje dentro de las limitaciones de recursos del dispositivo integrado, el cual, a continuación, devuelve la parte al cliente;

añadir la parte devuelta a cualesquiera partes previas recibidas en el cliente (104); y

repetir las etapas previas de envío y adición hasta que el archivo se haya reconstruido completamente en el cliente (104);

en donde el cliente (104) accede al archivo reconstruido como un proxy para el dispositivo integrado.

2. Método según la reivindicación 1, en el que los protocolos de enlace de comunicaciones se seleccionan del grupo consistente en HTTP, FTP, y cualquier combinación de los mismos.

3. Método según la reivindicación 1, en el que el archivo reconstruido comprende contenido que es por lo menos uno de procesado, transformado, manipulado, y acumulado en el cliente.

4. Método según la reivindicación 1, que comprende además la etapa de usar una pluralidad de enlaces de comunicación para acceder simultáneamente a contenido de múltiples archivos.

5. Método según la reivindicación 1, en el que el enlace de comunicaciones comprende una primera conexión que envía la parte del archivo y una segunda conexión que devuelve la parte.

6. Método según la reivindicación 5, que comprende además la etapa de usar una ventana de TCP para proporcionar un control de flujo automático entre la primera conexión y la segunda conexión.

7. Método según la reivindicación 1, en el que el método se utiliza para aplicaciones de gestión de rendimientos de fabricación.

8. Método para permitir que un dispositivo integrado (102) funcione conjuntamente con un cliente (104) en un ordenador anfitrión (106), en donde el cliente tiene acceso limitado a sistemas de archivos accesibles para el ordenador anfitrión, con el fin de generar un archivo en el cliente (104) a almacenar o abrir a través del ordenador anfitrión (106), que comprende las etapas siguientes:

generar contenido de un archivo en una memoria del cliente (104);

establecer un enlace de comunicaciones entre el cliente (104) y el dispositivo integrado (102);

determinar una de las siguientes opciones en el cliente (104): dónde se almacenará el archivo en el sistema de archivos accesible para el ordenador anfitrión y si se abrirá el archivo;

enviar una parte del archivo desde el cliente al dispositivo integrado de un tamaño que encaje dentro de las limitaciones de recursos del dispositivo integrado (102), el cual devuelve la parte de nuevo al cliente (104),

añadir la parte devuelta a cualesquiera partes previas recibidas en el ordenador anfitrión, y repetir las etapas previas de envío y adición hasta que el archivo completo se haya enviado y devuelto completamente; y,

o bien almacenar o bien abrir el archivo completo en concordancia con la etapa de determinación.

9. Método según la reivindicación 8, en el que los protocolos de enlace de comunicaciones se seleccionan del grupo consistente en HTTP, FTP, y cualquier combinación de los mismos.

10. Método según la reivindicación 8, en el que la etapa de generar contenido comprende generar contenido en tiempo real a partir de datos a medida que esos datos son proporcionados por el dispositivo integrado (102).

11. Método según la reivindicación 8, en el que la etapa de generar contenido comprende procesar información recibida por el cliente (104) desde el dispositivo integrado (102) de una de las siguientes maneras: transformación, manipulación, acumulación, y cualquier combinación de las mismas.

12. Método según la reivindicación 8, que comprende además la etapa de usar una pluralidad de enlaces de comunicación para enviar y devolver simultáneamente múltiples archivos.

13. Método según la reivindicación 8, en el que el enlace de comunicaciones comprende una primera conexión que envía la parte del archivo y una segunda conexión que devuelve la parte.

14. Método según la reivindicación 13, que comprende además la etapa de usar una ventana de TCP para proporcionar un control de flujo automático entre la primera conexión y la segunda conexión.

15. Método según la reivindicación 8, en el que el método se utiliza para aplicaciones de gestión de rendimientos de fabricación.

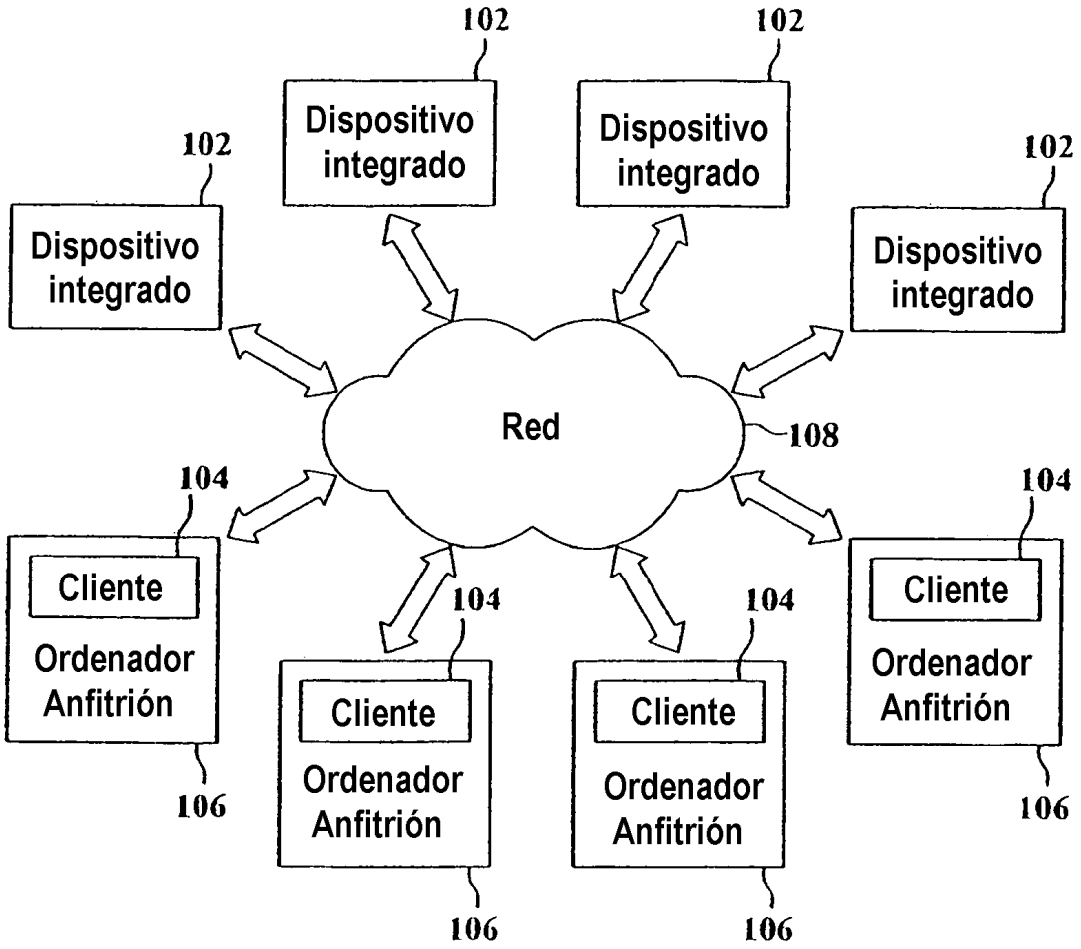

102
Dispositivo integrado
102
Dispositivo integrado
102
Dispositivo integrado
102
Dispositivo integrado
Red
108
104
Cliente
Ordenador Anfitrión
106
104
Cliente
Ordenador Anfitrión
106
104
Cliente
Ordenador Anfitrión
106
104
Cliente
Ordenador Anfitrión
106

FIG. 1

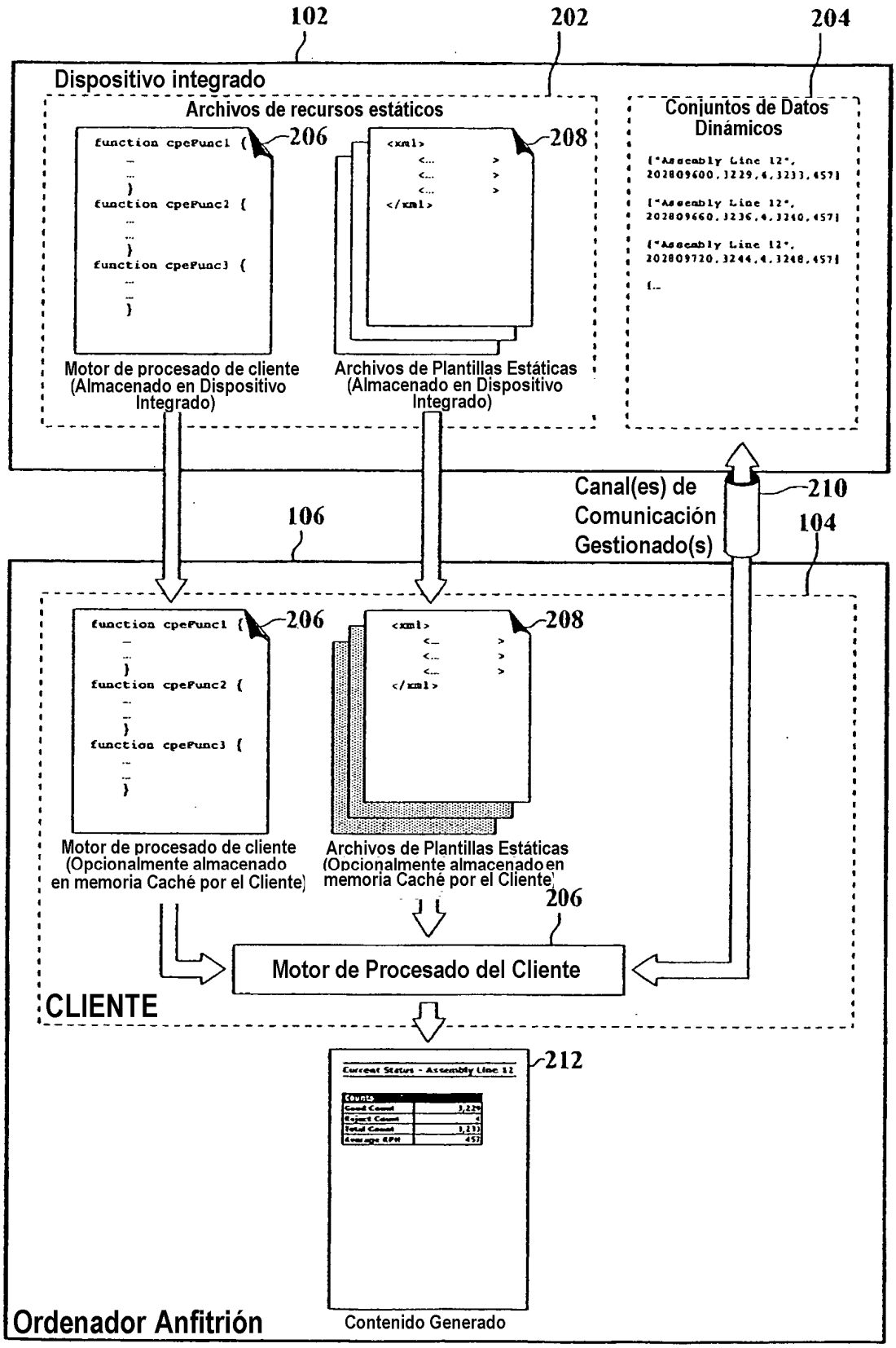

102
202
204
Dispositivo integrado
Archivos de recursos estáticos
Conjuntos de Datos Dinámicos
206
208
Motor de procesado de cliente (Almacenado en Dispositivo Integrado)
Archivos de Plantillas Estáticas (Almacenado en Dispositivo Integrado)
Canal(es) de Comunicación Gestionado(s)
210
106
104
Motor de procesado de cliente (Opcionalmente almacenado en memoria Caché por el Cliente)
Archivos de Plantillas Estáticas (Opcionalmente almacenado en memoria Caché por el Cliente)
Motor de Procesado del Cliente
CLIENTE
212
Ordenador Anfitrión
Contenido Generado

FIG. 2

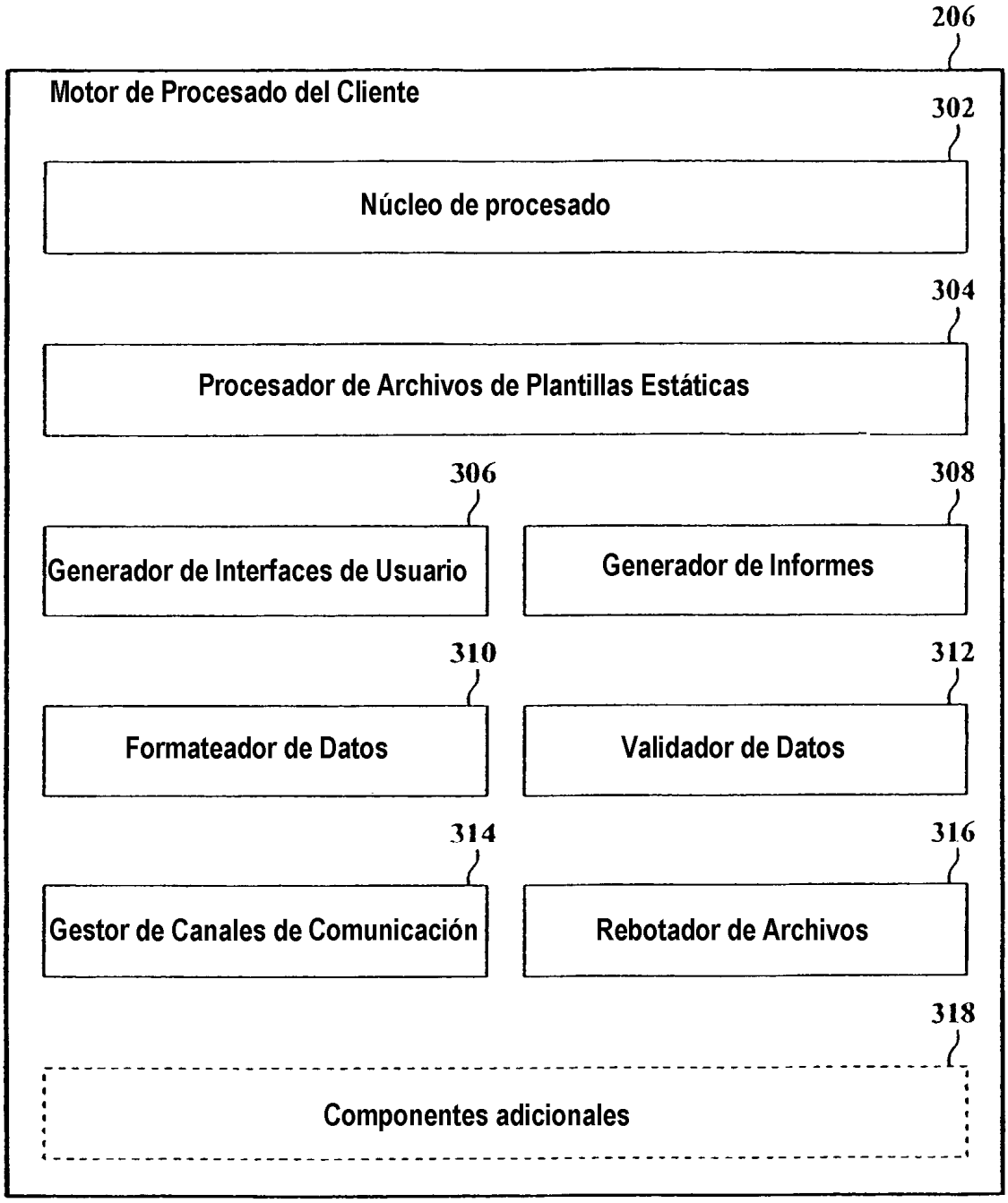

206
Motor de Procesado del Cliente
302
Núcleo de procesado
304
Procesador de Archivos de Plantillas Estáticas
306
Generador de Interfaces de Usuario
308
Generador de Informes
310
Formateador de Datos
312
Validador de Datos
314
Gestor de Canales de Comunicación
316
Rebotador de Archivos
318
Componentes adicionales

FIG. 3

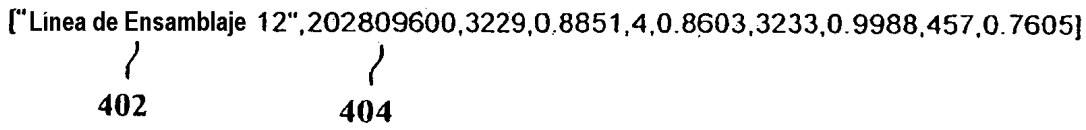

["Linea de Ensamblaje 12",202809600,3229,0.8851,4,0.8603,3233,0.9988,457,0.7605]
402
404

FIG. 4

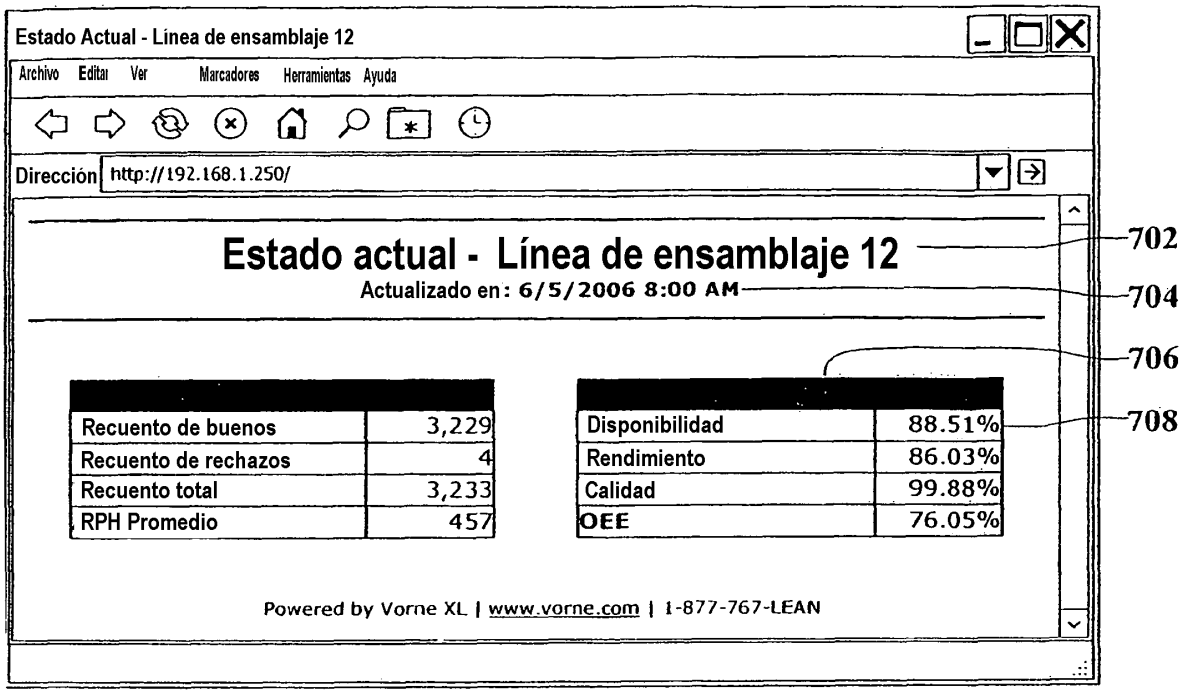
```
<?xml version="1.0" encoding="UTF-8" ?>
<static_template_file>
    <header>                    502      504
        <title index="0" label=" estado actual  - " format="@" />
        <subtitle index="1" label="Actualizado en : " format="m/d/yyyy h:mm AM/PM" />
    </header>
    <stack_panel type="horizontal">
      <variable_box label="Recuentos">
            <variable index="2" label="Recuento de buenos " format="#,0" />
510         <variable index="4" label="Recuento de rechazos " format="#,0" />
            <variable index="6" label="Recuento total " format="#,0" />
            <variable index="8" label="RPH promedio " format="#,0" />
        </variable_box>
        <variable_box label="OEE">
            <variable index="3" label="Disponibilidad" format="#,0.00%" />
            <variable index="5" label="Rendimiento" format="#,0.00%" />
512         <variable index="7" label="Calidad" format="#,0.00%" />
            <variable index="9" label="OEE" format="#,0.00%" />
508     </variable_box>                                    506
    </stack_panel>
    <footer />
</static_template_file>
```

FIG. 5

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title> Estado actual - Assembly Line 12</title>
        <style type="text/css">
            body {font-family: "verdana"; width: 720px;}
            table {border-collapse: collapse; font-size: 10pt;}
            td {border: 0.5pt black solid;}
            thead {background-color: black; color: white; font-weight: bold;}
            .horizontal_stack_panel_element {float: left;}
            .name {font-weight: bold;}
            .stack_panel_final_element {clear: both;}
            .value {text-align: right;}
            #header {border-bottom: 2pt black solid; border-top: 2pt black solid;
                margin-bottom: 24pt; text-align: center;}
            #header h1 {font-size: 18pt; margin: 2pt 0pt 0pt;}
            #header h2 {font-size: 9pt; margin: 0pt 0pt 6pt;}
            #footer {font-size: 8pt; margin-top: 20pt; text-align: center;}
        </style>
    </head>
    <body>
        <div id="header"><h1> Estado actual - Assembly Line 12</h1>
            <h2> Actualizado en : 6/5/2006 8:00 AM</h2></div>
        <div class="horizontal_stack_panel">
            <div style="width: 50%;" class="horizontal_stack_panel_element">
                <table style="margin: auto; width: 300px;">
                    <thead><tr><td>Recuentos </td><td></td></tr></thead>
                    <tbody>
                        <tr><td class="name">Recuento de buenos </td><td class="value">3,229</td></tr>
                        <tr><td class="name">Recuento de rechazos </td><td class="value">4</td></tr>
                        <tr><td class="name">Recuento total </td><td class="value">3,233</td></tr>
                        <tr><td class="name">RPH promedio </td><td class="value">457</td></tr>
                    </tbody>
                </table>
            </div>
            <div style="width: 50%;" class="horizontal_stack_panel_element">
                <table style="margin: auto; width: 300px;">
                    <thead><tr><td>OEE</td><td></td></tr></thead>
                    <tbody>
                        <tr><td class="name">Disponibilidad </td><td class="value">88.51%</td></tr>
                        <tr><td class="name">Rendimiento  </td><td class="value">86.03%</td></tr>
                        <tr><td class="name">Calidad </td><td class="value">99.88%</td></tr>
                        <tr><td class="name">OEE</td><td class="value">76.05%</td></tr>
                    </tbody>
                </table>
            </div>
            <div class="stack_panel_final_element"></div>
        </div>
        <div id="footer"> Powered by Vorne XL | <a href="http://www.vorne.com">www.vorne.com</a> |
            1-877-767-LEAN </div>
    </body>
</html>
```

602

FIG. 6



Archivo Editar Ver Marcadores Herramientas Ayuda

FIG. 7

```
<?xml version="1.0" encoding="UTF-8" ?>
<static_template_file type="rtf">
    <header>{\rtf1\ansi\deff0{\fonttbl{\f0\fswiss Verdana}}
        {\colortbl ;\red0\green0\blue0;\red255\green255\blue255;}
        \margt720\margb720\margl1080\margr1080
802 </header>
    <cuerpo_registro >
        \pard\brdrt\brdrs\brdrw40\brdrcf1\brdrb\brdrs\brdrw40\brdrcf1\qc
        {\b\fs4 \~\line\fs36 Shift Report - <variable format="@"/>
808     \fs20\line Shift Start: <variable format="m/d/yyyy h:mm AM/PM"/>
        \fs12\line \~}\par\pard\fs20\par\par\par{\trowd \trgaph45
        <define text="cell">
810     \clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs
        </define>
        \clcbpat1\cellx2520
        <insert text="cell"/>\clcbpat1\cellx4680\cellx5400
812     <insert text="cell"/>\clcbpat1\cellx7920
        <insert text="cell"/>\clcbpat1\cellx10080
        \pard\intbl\cf2\ql {\b Recuentos }\cell\pard\intbl\cf2\qr {}\cell
        \pard\intbl\cf2\ql {}\cell
        \pard\intbl\cf2\ql {\b OEE}\cell\pard\intbl\cf2\qr {<define text="row">}\cell
        \row}{\trowd \trgaph45
        <insert text="cell"/>\cellx2520
        <insert text="cell"/>\cellx4680\cellx5400
        <insert text="cell"/>\cellx7920
        <insert text="cell"/>\cellx10080                                    814
        </define>                                                           /
        \pard\intbl\cf1\ql {\b Recuento de buenos }\cell\pard\intbl\cf1\qr {<variable format="#,0"/>}\cell
        \pard\intbl\cf1\ql {}\cell
        \pard\intbl\cf1\ql {\b Disponibilidad}\cell\pard\intbl\cf1\qr {<variable format="#,0.00%"/><insert text="row"/>
        \pard\intbl\cf1\ql {\b Recuento de rechazos }\cell\pard\intbl\cf1\qr {<variable format="#,0"/>}\cell
        \pard\intbl\cf1\ql {}\cell
        \pard\intbl\cf1\ql {\b Rendimiento }\cell\pard\intbl\cf1\qr {<variable format="#,0.00%"/><insert text="row"/>
        \pard\intbl\cf1\ql {\b Recuento total }\cell\pard\intbl\cf1\qr {<variable format="#,0"/>}\cell
        \pard\intbl\cf1\ql {}\cell
        \pard\intbl\cf1\ql {\b Calidad }\cell\pard\intbl\cf1\qr {<variable format="#,0.00%"/><insert text="row"/>
        \pard\intbl\cf1\ql {\b RPH promedio }\cell\pard\intbl\cf1\qr {<variable format="#,0"/>}\cell
        \pard\intbl\cf1\ql {}\cell                                                          506
        \pard\intbl\cf1\ql {\b OEE}\cell\pard\intbl\cf1\qr {<variable format="#,0.00%"/>}\cell
        \row}
        \pard\par\par\pard\fs16\cf1\qc Powered by Vorne XL | {\field{\*\fldinst{HYPERLINK
806     "http://www.vorne.com"}}{\fldrslt{\cf1\ul www.vorne.com}}}\cf1  | 1-877-767-LEAN\par
    </cuerpo_registro >
    <separador_registros >\page</ separador_registros >
    <footer>}</footer>
</static_template_file>                                                  804
```

FIG. 8

{\rtf1\ansi\deff0{\fonttbl{\f0\fswiss Verdana}}
{\colortbl ;\red0\green0\blue0;\red255\green255\blue255;}
\margt720\margb720\margl1080\margr1080
\pard\brdrt\brdrs\brdrw40\brdrcf1\brdrb\brdrs\brdrw40\brdrcf1\qc
{\b\fs4 \~\line\fs36 Shift Report - Assembly Line 12
\fs20\line Shift Start: 6/5/2006 8:00 AM
\fs12\line \~}\par\pard\fs20\par\par\par{\trowd \trgaph45
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\clcbpat1\cellx2520
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\clcbpat1\cellx4680
\cellx5400
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\clcbpat1\cellx7920
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\clcbpat1\cellx10080
\pard\intbl\cf2\ql {\b Recuentos }\cell\pard\intbl\cf2\qr {}\cell
\pard\intbl\cf2\ql {}\cell
\pard\intbl\cf2\ql {\b OEE}\cell\pard\intbl\cf2\qr {}\cell
\row}{\trowd \trgaph45
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx2520
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx4680\cellx5400
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx7920
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx10080
\pard\intbl\cf1\ql {\b Recuento de buenos }\cell\pard\intbl\cf1\qr {3,229}\cell
\pard\intbl\cf1\ql {}\cell
\pard\intbl\cf1\ql {\b Disponibilidad }\cell\pard\intbl\cf1\qr {88.51%}\cell
\row}{\trowd \trgaph45
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx2520
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx4680\cellx5400
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx7920
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx10080
\pard\intbl\cf1\ql {\b Recuento de rechazos }\cell\pard\intbl\cf1\qr {4}\cell
\pard\intbl\cf1\ql {}\cell
\pard\intbl\cf1\ql {\b Rendimiento }\cell\pard\intbl\cf1\qr {86.03%}\cell
\row}{\trowd \trgaph45
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx2520
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx4680\cellx5400
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx7920
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx10080
\pard\intbl\cf1\ql {\b Recuento total }\cell\pard\intbl\cf1\qr {3,233}\cell
\pard\intbl\cf1\ql {}\cell
\pard\intbl\cf1\ql {\b Calidad }\cell\pard\intbl\cf1\qr {99.88%}\cell
\row}{\trowd \trgaph45
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx2520
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx4680\cellx5400
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx7920
\clbrdrt\brdrw10\brdrs\clbrdrl\brdrw10\brdrs\clbrdrb\brdrw10\brdrs\clbrdrr\brdrw10\brdrs\cellx10080
\pard\intbl\cf1\ql {\b RPH promedio}\cell\pard\intbl\cf1\qr {457}\cell
\pard\intbl\cf1\ql {}\cell
\pard\intbl\cf1\ql {\b OEE}\cell\pard\intbl\cf1\qr {76.05%}\cell\row}
\pard\par\par\pard\fs16\cf1\qc Powered by Vorne XL | {\field{*\fldinst{HYPERLINK
"http://www.vorne.com"}}{\fldrslt{\cf1\ul www.vorne.com}}}\cf1 | 1-877-767-LEAN\par}

Rendimiento Calidad

FIG. 9

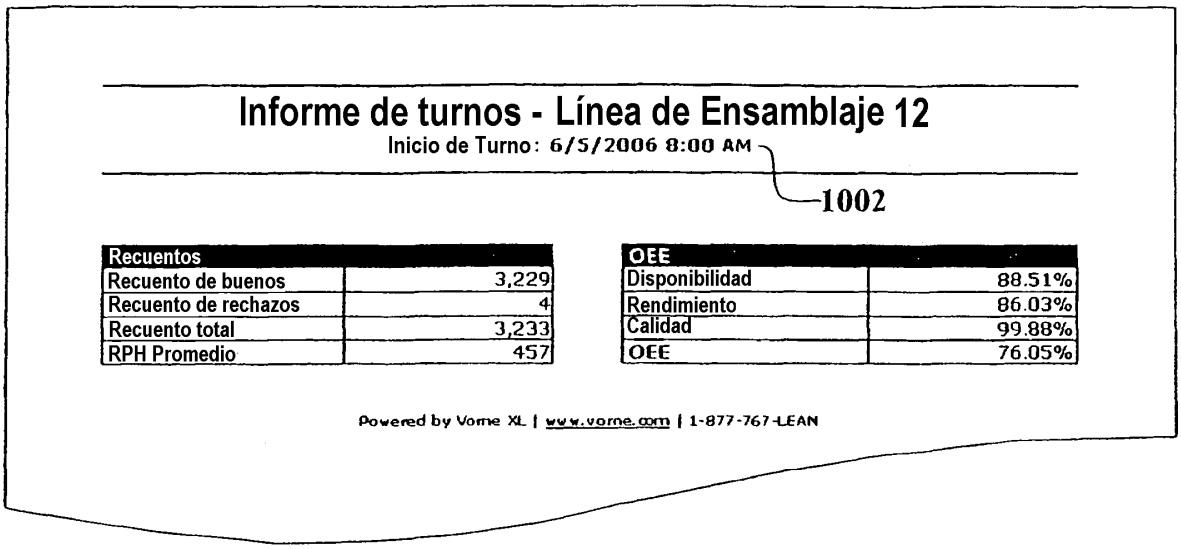

Informe de turnos - Línea de Ensamblaje 12
Inicio de Turno: 6/5/2006 8:00 AM
1002
Recuentos
Recuento de buenos 3,229
Recuento de rechazos 4
Recuento total 3,233
RPH Promedio 457
OEE
Disponibilidad 88.51%
Rendimiento 86.03%
Calidad 99.88%
OEE 76.05%
Powered by Vorne XL | www.vorne.com | 1-877-767-LEAN

FIG. 10

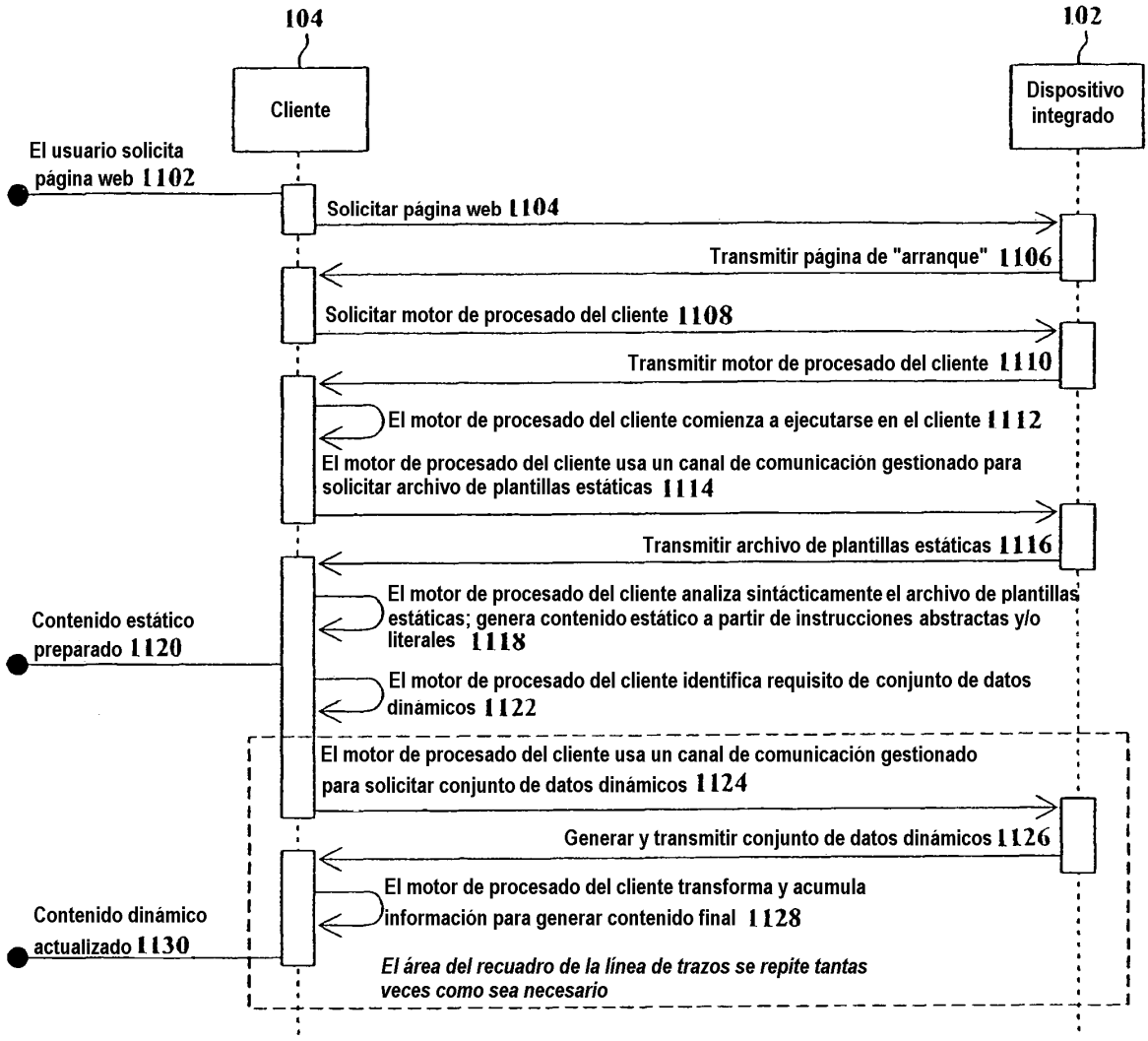

104
Cliente
102
Dispositivo integrado
El usuario solicita página web 1102
Solicitar página web 1104
Transmitir página de "arranque" 1106
Solicitar motor de procesado del cliente 1108
Transmitir motor de procesado del cliente 1110
El motor de procesado del cliente comienza a ejecutarse en el cliente 1112
El motor de procesado del cliente usa un canal de comunicación gestionado para solicitar archivo de plantillas estáticas 1114
Transmitir archivo de plantillas estáticas 1116
El motor de procesado del cliente analiza sintácticamente el archivo de plantillas estáticas; genera contenido estático a partir de instrucciones abstractas y/o literales 1118
Contenido estático preparado 1120
El motor de procesado del cliente identifica requisito de conjunto de datos dinámicos 1122
El motor de procesado del cliente usa un canal de comunicación gestionado para solicitar conjunto de datos dinámicos 1124
Generar y transmitir conjunto de datos dinámicos 1126
El motor de procesado del cliente transforma y acumula información para generar contenido final 1128
Contenido dinámico actualizado 1130
El área del recuadro de la línea de trazos se repite tantas veces como sea necesario

FIG. 11

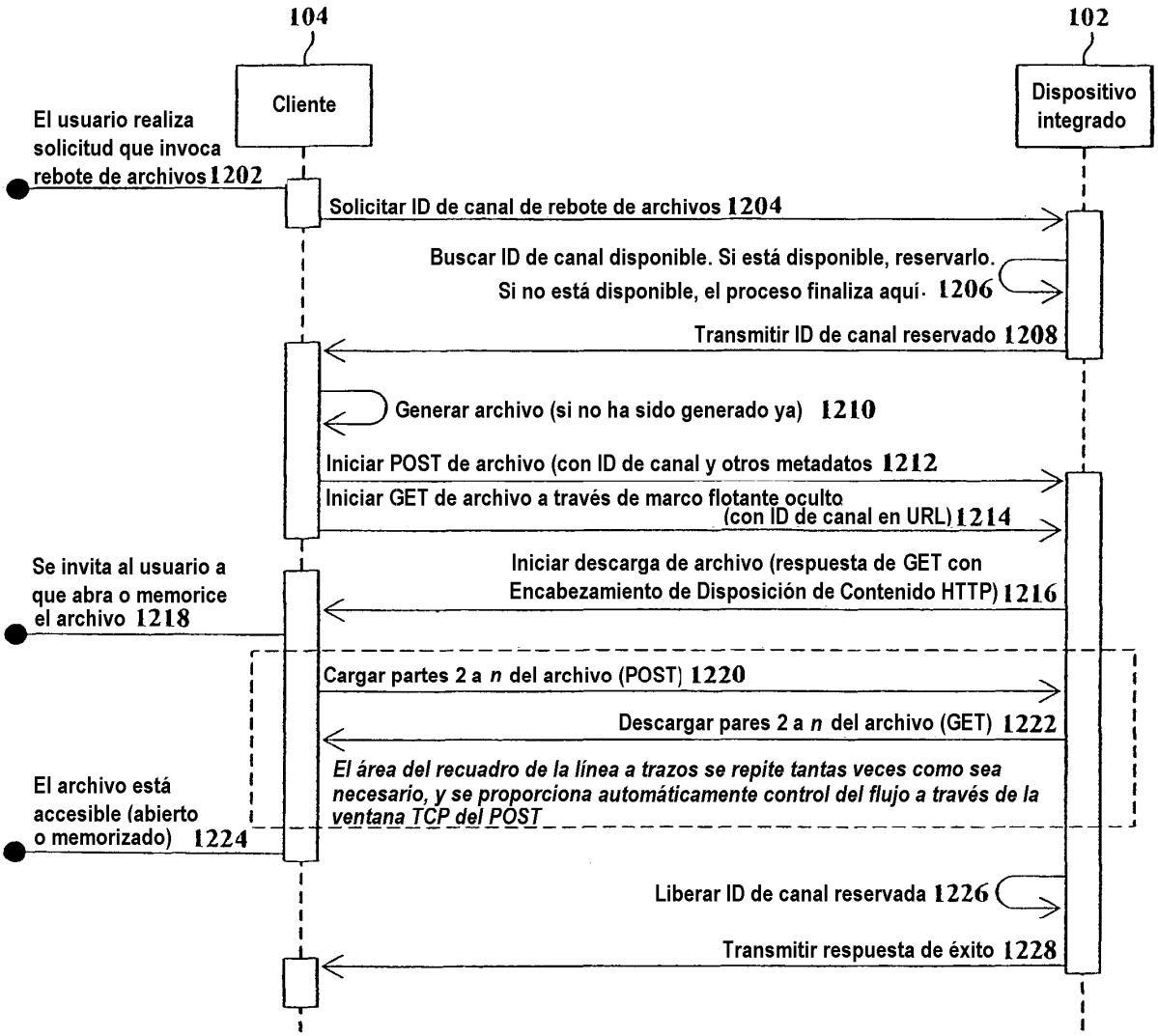

104
Cliente
102
Dispositivo integrado
El usuario realiza solicitud que invoca rebote de archivos 1202
Solicitar ID de canal de rebote de archivos 1204
Buscar ID de canal disponible. Si está disponible, reservarlo.
Si no está disponible, el proceso finaliza aquí. 1206
Transmitir ID de canal reservado 1208
Generar archivo (si no ha sido generado ya) 1210
Iniciar POST de archivo (con ID de canal y otros metadatos 1212
Iniciar GET de archivo a través de marco flotante oculto (con ID de canal en URL) 1214
Se invita al usuario a que abra o memorice el archivo 1218
Iniciar descarga de archivo (respuesta de GET con Encabezamiento de Disposición de Contenido HTTP) 1216
Cargar partes 2 a n del archivo (POST) 1220
Descargar pares 2 a n del archivo (GET) 1222
El área del recuadro de la línea a trazos se repite tantas veces como sea necesario, y se proporciona automáticamente control del flujo a través de la ventana TCP del POST
El archivo está accesible (abierto o memorizado) 1224
Liberar ID de canal reservada 1226
Transmitir respuesta de éxito 1228

FIG. 12

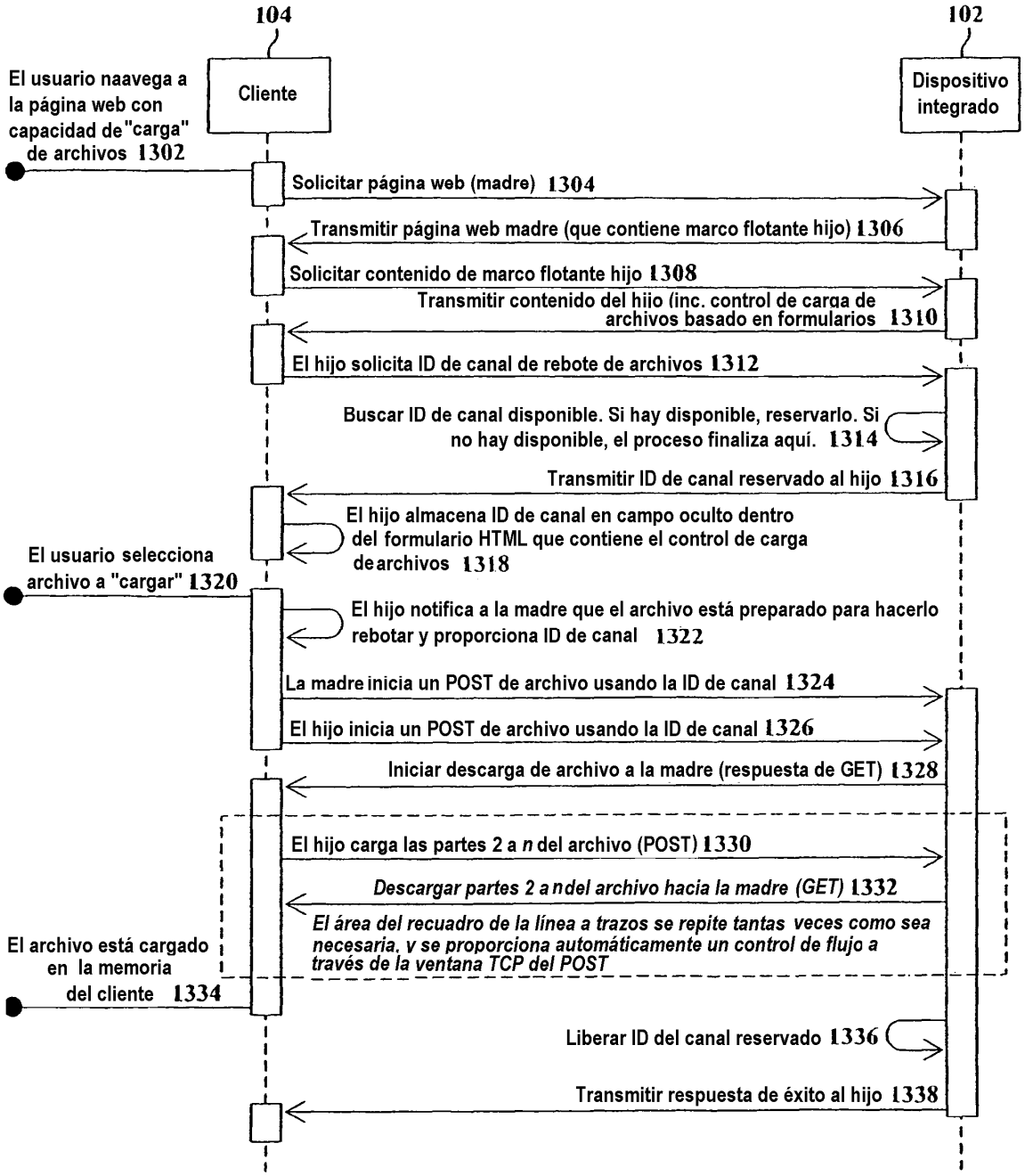

104
102
Cliente
Dispositivo integrado
El usuario naavega a la página web con capacidad de "carga" de archivos 1302
Solicitar página web (madre) 1304
Transmitir página web madre (que contiene marco flotante hijo) 1306
Solicitar contenido de marco flotante hijo 1308
Transmitir contenido del hijo (inc. control de carga de archivos basado en formularios 1310
El hijo solicita ID de canal de rebote de archivos 1312
Buscar ID de canal disponible. Si hay disponible, reservarlo. Si no hay disponible, el proceso finaliza aquí. 1314
Transmitir ID de canal reservado al hijo 1316
El hijo almacena ID de canal en campo oculto dentro del formulario HTML que contiene el control de carga de archivos 1318
El usuario selecciona archivo a "cargar" 1320
El hijo notifica a la madre que el archivo está preparado para hacerlo rebotar y proporciona ID de canal 1322
La madre inicia un POST de archivo usando la ID de canal 1324
El hijo inicia un POST de archivo usando la ID de canal 1326
Iniciar descarga de archivo a la madre (respuesta de GET) 1328
El hijo carga las partes 2 a n del archivo (POST) 1330
Descargar partes 2 a n del archivo hacia la madre (GET) 1332
El área del recuadro de la línea a trazos se repite tantas veces como sea necesaria, y se proporciona automáticamente un control de flujo a través de la ventana TCP del POST
El archivo está cargado en la memoria del cliente 1334
Liberar ID del canal reservado 1336
Transmitir respuesta de éxito al hijo 1338

FIG. 13