(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 612 110**

(51) Int. CI.:

G06F 9/44 (2006.01)
G06F 9/46 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **14.05.2001 E 01111682 (9)**

(97) Fecha y número de publicación de la concesión europea: **26.10.2016 EP 1164473**

(54) Título: **Gestión de estados de objetos de control de lado de servidor**

(30) Prioridad:

18.05.2000 US 574144

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

(73) Titular/es:

MICROSOFT TECHNOLOGY LICENSING, LLC (100.0%)
One Microsoft Way
Redmond, WA 98052, US

(72) Inventor/es:

BURD, GARY S.;
COOPER, KENNETH B.;
ANDERS, MARK T.;
GUTHRIE, SCOTT D.;
EBBO, DAVID S.;
PETERS, TED A. y
MILLET, STEPHEN J.

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

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

DESCRIPCIÓN

Gestión de estados de objetos de control de lado de servidor

La invención se refiere en general a una estructura de servidor web y, más en particular, a la gestión del estado de objetos de control de lado de servidor que procesan elementos de interfaz de usuario de lado de cliente de una página web.

Un navegador web típico recibe datos desde un servidor web que define la apariencia y el comportamiento elemental de una página web para la representación en un sistema de cliente. En un escenario típico, un usuario especifica un localizador uniforme de recursos ("URL", *Uniform Resource Locator*), que es una dirección global de un recurso de la Red Mundial, para acceder a un sitio web deseado. Por lo general, el término "recurso" se refiere a datos o rutinas a los que se puede acceder mediante un programa. Un ejemplo de URL es "http://www.microsoft.com/ms.htm". La primera parte del ejemplo de URL indica un protocolo dado (es decir, "http") que se va a usar en la comunicación. La segunda parte especifica el nombre de dominio (es decir,"www.microsoft.com") en el que está ubicado el recurso. La tercera parte especifica el recurso (es decir, un archivo llamado "ms.htm") dentro del dominio. De forma correspondiente, un navegador genera una solicitud de HTTP (protocolo de transporte de hipertexto) asociada al ejemplo de URL para obtener los datos asociados al archivo ms.htm dentro del dominio www.microsoft.com. Un servidor web que aloja el sitio www.microsoft.com recibe la solicitud de HTTP y devuelve la página web o recurso solicitado en una respuesta de HTTP al sistema de cliente para la representación en el navegador.

El archivo "ms.htm" del ejemplo en lo que antecede incluye código de HTML (*HyperText Markup Language*, lenguaje de marcado de hipertexto) estático. El lenguaje HTML es un lenguaje de creación de texto sin formato que se usa para crear documentos (por ejemplo, páginas web) en la Red Mundial. En ese sentido un archivo de HTLM puede ser recuperado desde un servidor web y representado como una página web en un navegador. Usando HTML, un desarrollador puede, por ejemplo, especificar texto con formato, listas, formularios, tablas, enlaces de hipertexto, imágenes y sonidos incorporados, y gráficos de fondo para la representación en el navegador para presentar la rica experiencia gráfica que los usuarios han llegado a esperar mientras ven información de Internet. No obstante, un archivo de HTML es un archivo estático que no soporta de forma inherente la generación dinámica de contenido de página web.

Si se va a representar un contenido dinámico, tal como un cambio en la cotización de acciones o información de tráfico, un programa de aplicación de lado de servidor se desarrolla, en general, para manejar la interacción cliente - servidor, más compleja. El programa de aplicación de lado de servidor procesa una solicitud de HTTP y genera el código de HTML apropiado para la transmisión al cliente en una respuesta de HTTP. Una solicitud de HTTP a modo de ejemplo puede incluir parámetros, tales como datos en una cadena de consulta o datos a partir de formularios basados en web. En ese sentido, un programa de aplicación de lado de servidor puede procesar los parámetros y generar de forma dinámica un código de HTML en una respuesta de HTTP al cliente. Un programa de aplicación de lado de servidor a modo de ejemplo puede generar, de forma dinámica, documentos que contienen un código de HTML apropiado usando una secuencia de una o más operaciones de escritura de texto con formato en una estructura de memoria. A continuación de lo anterior, el documento resultante se transmite a un sistema de cliente en una respuesta de HTTP, en la que este se representa como una página web en el navegador.

El desarrollo de un programa de aplicación de lado de servidor puede ser una tarea compleja que requiera familiaridad no solo con la codificación de HTML normal que se usa para configurar una página web, sino también con nociones elementales de programación, incluyendo uno o más lenguajes de programación (por ejemplo, C++, Perl, Visual Basic o Jscript). Los diseñadores de páginas web, por otra parte, son frecuentemente diseñadores gráficos y editores, que pueden carecer de experiencia en programación. Asimismo, la simplificación del complejo desarrollo de una página web puede acelerar el desarrollo de nuevo contenido web por parte de cualquier desarrollador. Por lo general, el desarrollo de un programa de aplicación de lado de servidor habitual también requiere un gran esfuerzo, tanto, de hecho, que los desarrolladores son frecuentemente reacios a intentarlo. En consecuencia, es deseable proporcionar una estructura de desarrollo que permita a un desarrollador crear y procesar de forma dinámica una página web con una programación mínima.

Un enfoque para minimizar los requisitos de programación de la generación de páginas web dinámicas ha sido la estructura de Página de Servidor Activo (ASP, *Active Server Page*), proporcionada por Microsoft Corporation. Un recurso de ASP incluye normalmente código Visual Basic o Jscript, por ejemplo, para procesar una solicitud de HTTP que especifique el recurso de ASP como el recurso deseado y, tras ello, para generar el código de HTML resultante en una respuesta de HTTP al cliente. Asimismo, un recurso de ASP puede hacer referencia a componentes de biblioteca de lado de cliente de terceras partes o previamente desarrollados (por ejemplo, controles de ACTIVEX de lado de cliente) en lugar de requerir que el desarrollador de la página escriba todos los componentes desde cero. Sin embargo, en las estructuras actuales de aplicación de lado de servidor, la programación requerida para gestionar de manera dinámica los elementos de interfaz de usuario de lado de cliente (por ejemplo, cuadros de texto, cuadros de lista, botones, enlaces de hipertexto, imágenes, sonidos, etc.) dentro de las aplicaciones de lado de servidor pueden seguir requiriendo capacidades de programación sofisticadas y un esfuerzo considerable. Existe un problema sin respuesta en el encapsulamiento adecuado de la programación requerida para procesar elementos de interfaz de usuario, de manera que se permita al desarrollador de páginas

web centrarse en otros aspectos de la página web.

El procesamiento de lado de servidor de los elementos de interfaz de usuario de lado de cliente, no obstante, puede implicar complejas cuestiones de gestión de estados que, por lo general, corresponderían al programa de aplicaciones de lado de servidor. El estado de un módulo de procesamiento de lado de servidor que se corresponde con el elemento de interfaz de usuario de lado de cliente representa la configuración y las propiedades del módulo de lado de servidor. Un modelo de cliente / servidor sin estados, no obstante, indica que, en las comunicaciones entre el cliente y el servidor, el servidor no mantiene el estado de los procedimientos que tienen lugar entre el cliente y el servidor, debido a que la conexión de comunicaciones entre el cliente y el servidor puede desaparecer de forma inesperada.

El documento US-A-5 991 802 expone un procedimiento y sistema para invocar, mediante un sistema informático de cliente, una función de un objeto de una clase de objetos proporcionada por un sistema informático de servidor. El cliente envía una solicitud a un servidor que comprende un URL que identifica una secuencia de comandos, una clase de objetos y una función de la clase de objetos que han de invocarse. En respuesta a la recepción de la solicitud, el servidor comienza la secuencia de comandos y transfiere el control a la secuencia de comandos, el cual instancia un objeto de la clase de objetos identificada en el URL de la solicitud recibida e invoca la función identificada en el URL. La función invocada lleva a cabo el comportamiento de la función, crea una respuesta y envía la respuesta al cliente. La respuesta contiene una información de estado que describe un estado del objeto después de que se realice el comportamiento. Esta información de estado se puede incluir entonces en una solicitud adicional de tal modo que la función puede realizar su comportamiento basándose en la información de estado.

El objeto de la presente invención es proporcionar un procedimiento mejorado para gestionar un estado de un objeto de control de lado de servidor, así como un producto informático correspondiente.

Este objeto se soluciona por medio de la materia objeto de las reivindicaciones independientes.

Se definen realizaciones preferidas por medio de las reivindicaciones dependientes.

De acuerdo con la presente invención, el problema en lo que antecede, y otros problemas, son resueltos mediante la provisión de una gestión de estados encapsulada para el procesamiento de lado de servidor de los elementos de interfaz de usuario de lado de cliente en los que el servidor no mantiene el estado entre solicitudes. La operación de procesamiento del elemento de interfaz de usuario de lado de cliente puede incluir una operación de gestión de estados, que se refiere al estado (es decir, el “estado de vista”) de uno o más objetos de control de lado de servidor en una jerarquía de objetos de control. Un objeto de control de lado de servidor procesa y genera un elemento de interfaz de usuario de lado de cliente para la representación en una página web. Una jerarquía de objetos de control de lado de servidor también puede cooperar para generar el código de lenguaje de creación resultante, tal como la norma de HTML, para la representación de una página web en un cliente. El cliente puede ser, por ejemplo, cualquier navegador que soporte la norma de HTML u otro lenguaje de creación.

Para satisfacer el modelo de cliente / servidor sin estados, la información de estado de los objetos de control de lado de servidor se puede transportar entre el cliente y el servidor en una estructura de estado transportable, en lugar de permanecer en el servidor entre solicitudes de HTTP. Cuando el servidor recibe una solicitud de HTTP a partir del cliente, el servidor extrae la información de estado a partir de una estructura de estado transportable a partir de la solicitud de HTTP y distribuye la información de estado a los objetos de control individuales apropiados en la jerarquía de lado de servidor. La integridad de la estructura de estado transportable también se puede verificar usando una estructura codificada que está asociada con la estructura de estado transportable y que se genera a partir de la información de estado.

Se proporciona un procedimiento para gestionar un estado de un objeto de control de lado de servidor que se corresponde con un elemento de interfaz de usuario de lado de cliente que está incorporado en una página web que se representa en un cliente. El objeto de control de lado de servidor se crea en la jerarquía de objetos de control para procesar el elemento de interfaz de usuario de lado de cliente. Una estructura de estado transportable se recibe a partir del cliente. La estructura de estado transportable incluye una información de estado que indica un valor de estado para al menos un objeto de control de lado de servidor en la jerarquía de objetos de control. La información de estado se extrae a partir de la estructura de estado transportable. El valor de estado a partir de la información de estado se carga en la propiedad del objeto de control de lado de servidor, si el valor de estado está asociado con el objeto de control de lado de servidor.

Se proporciona un producto de programa informático para gestionar un estado de una pluralidad de objetos de control de lado de servidor que se crea en una jerarquía de objetos de control y que se corresponde con una pluralidad de elementos de interfaz de usuario de lado de cliente que están incorporados en una página web que se representa en un cliente. Una estructura de estado transportable se recibe a partir del cliente incluye una información de estado que está asociada con uno o más objetos de control de lado de servidor en la jerarquía de objetos de control. La información de estado se serializa para extraer un valor de estado, un tipo de datos de propiedad asociado y una información jerárquica para una propiedad de un objeto de control de lado de servidor. El objeto de control de lado de servidor se ubica dentro de la jerarquía de objetos de control basándose en la información

jerárquica. El valor de estado se carga en la propiedad del objeto de control de lado de servidor.

Un artículo de fabricación se proporciona como un producto de programa informático. Una realización de un producto de programa informático de acuerdo con la presente invención incluye un medio de almacenamiento de programa informático legible por un sistema informático y que codifica un programa informático para ejecutar un procedimiento informático que gestiona un estado de un objeto de control de lado de servidor que se corresponde con un elemento de interfaz de usuario de lado de cliente que está incorporado en una página web que se representa en un cliente. Una realización alternativa de un producto de programa informático de acuerdo con la presente invención incluye una señal de datos informáticos que se materializa en una onda portadora por medio de un sistema informático y que codifica un programa informático que gestiona un estado de un objeto de control de lado de servidor que se corresponde con un elemento de interfaz de usuario de lado de cliente que está incorporado en una página web que se representa en un cliente. Un producto que es generado por un procedimiento de la presente invención se proporciona como un código de lenguaje de creación, que contiene una estructura de estado transportable, que se transmite a un cliente y que es interpretado por un navegador en el cliente.

Breve descripción de los dibujos

La figura 1 ilustra un servidor web para generar de forma dinámica contenido de página web para la representación en una realización de la presente invención.

La figura 2 ilustra un diagrama de flujo de operaciones para procesar y sintetizar elementos de interfaz de usuario de lado de cliente usando objetos de control de lado de servidor en una realización de la presente invención.

La figura 3 ilustra módulos de ejemplo en un servidor web que se usa en una realización de la presente invención.

La figura 4 ilustra un sistema a modo de ejemplo útil para implementar una realización de la presente invención.

La figura 5 ilustra un diagrama de flujo de procedimiento que representa el procesamiento de un objeto de página en una realización de la presente invención.

La figura 6 ilustra una porción a modo de ejemplo de un archivo de contenido dinámico (por ejemplo, un recurso de ASP+) en una realización de la presente invención.

La figura 7 ilustra un código resultante que es generado por uno o más objetos de control de lado de servidor en respuesta a la porción a modo de ejemplo del archivo de contenido dinámico de la figura 6.

La figura 8 ilustra la jerarquía de objetos de control que se corresponde con el recurso de ASP+ de la figura 6 y el valor de campo de _VIEWSTATE de la figura 7.

La figura 9 ilustra una versión anidada del valor de campo de _VIEWSTATE de la figura 7.

La figura 10 ilustra un diagrama de flujo de procedimiento para recibir una estructura de estado transportable y cargar la información de estado que está almacenada en la misma en los objetos de control de una jerarquía de objetos de control en una realización de la presente invención.

La figura 11 ilustra un diagrama de flujo de procedimiento para guardar información de estado a partir de los objetos de control de una jerarquía de objetos de control en una estructura de estado transportable para la transmisión al cliente en una realización de la presente invención.

Descripción detallada de la invención

En una realización de la presente invención, un contenido de página web se genera de forma dinámica en un servidor web para su representación en un cliente. El cliente y el servidor web se comunican a través de una red, por ejemplo, usando solicitudes de HTTP y respuestas de HTTP. En ese sentido, el servidor web genera un contenido de página web, que se puede encontrar en forma de código de HTML, y transmite el contenido al cliente, que puede representar la página web en un navegador. Los objetos de control de lado de servidor, que se corresponden de forma lógica con elementos de interfaz de usuario individuales de la página web, se crean en el servidor web para procesar y generar el contenido de página web que va a ser usado por un navegador de lado de cliente para representar y procesar una página web. Los objetos de control de lado de servidor se declaran en un archivo de contenido dinámico, tal como un recurso de ASP+, que es procesado por una fábrica de páginas para crear una jerarquía de objetos de control de lado de servidor. Los objetos de control en la jerarquía cooperan para procesar la solicitud que se recibe a partir del cliente y, a continuación, generan un contenido de página web resultante para la transmisión al cliente, antes de que se terminen los objetos de control en la jerarquía.

Un objeto de página se puede instanciar como el nivel superior de la jerarquía de objetos de control en una realización de la presente invención. Un objeto de página, que es también un objeto de control, por lo general contiene uno o más objetos de control hijos, y cada objeto de control hijo puede contener uno o más objetos de control hijos de sí mismo para extenderse a una jerarquía de múltiples niveles. Los objetos de página y los objetos de control descendente ejecutan una secuencia de operaciones para procesar y generar el contenido web que se corresponde con los elementos de interfaz de usuario de lado de cliente.

Una de las operaciones en esta secuencia incluye una operación de carga que deserializa la información de estado, que se recibe en una estructura de estado transportable que está incluida en la solicitud a partir del cliente, en el objeto u objetos de control apropiados en la jerarquía. Una operación de recorrido de esta operación de carga se puede desplazar a través de la jerarquía de objetos de control, siguiendo la información jerárquica en la estructura

de estado transportable, para ubicar unos objetos de control apropiados en la jerarquía. En una realización de la presente invención, la información de estado recibida incluye solo los estados de los objetos de control que han cambiado con respecto a sus estados iniciales (es decir, la información de estado incluye solo datos diferenciales). Realizaciones alternativas pueden incluir todos los estados de todos los objetos de control en la jerarquía, ya se hayan cambiado o no, o alguna otra combinación de información de estado e información jerárquica para uno o más objetos de control de lado de servidor.

Otra operación de la secuencia incluye una operación de guardado que serializa la información de estado a partir de los objetos de control que tienen estados cambiados. La información de estado se añade a una estructura de estado transportable para la transmisión al cliente. El cliente puede devolver la estructura de estado transportable al servidor web bajo una solicitud posterior al servidor web. Tras la recepción de la estructura de estado transportable, la jerarquía de objetos de control se recrea y la operación de carga restaura el estado de la jerarquía al de la respuesta previa.

En una realización alternativa, puede que la estructura de estado transportable no se transmita al cliente. En su lugar, la estructura de estado transportable puede permanecer en el servidor o transferirse a otra ubicación de recursos (por ejemplo, otro servidor). Por ejemplo, en el caso del equilibrado de carga, un primer servidor se puede comunicar con un navegador en una primera transacción de HTTP, pero un segundo servidor se puede comunicar con el navegador en una segunda transacción de HTTP. Por consiguiente, la estructura de estado transportable se puede comunicar desde el primer servidor al segundo servidor para procesar la segunda transacción de HTTP, sorteando de ese modo al cliente.

La figura 1 ilustra un servidor web para generar de forma dinámica contenido de página web para la representación en un cliente en una realización de la presente invención. Un cliente 100 ejecuta un navegador 102 que representa una página 104 web en un dispositivo de representación de cliente 100. El cliente 100 puede incluir un sistema informático de cliente que tenga un dispositivo de representación, tal como un monitor de vídeo. Un navegador “INTERNET EXPLORER”, comercializado por Microsoft Corporation, es un ejemplo de un navegador 102 en una realización de la presente invención. Otros ejemplos de navegadores incluyen sin limitación “NETSCAPE NAVIGATOR” y “MOSAIC”. El ejemplo de página 104 web incorpora un control de cuadro de texto 106 y dos controles de botón 108 y 110. El navegador 102 puede recibir código de HTML en la respuesta de HTTP 112 desde un servidor web 116 y representa la página web tal como se describe mediante el código de HTML. A pesar de que el lenguaje HTML es descrito con referencia a una realización, otros lenguajes de creación, incluyendo sin limitación el SGML (*Standard Generalized Markup Language*, lenguaje de marcado generalizado convencional), el XML (*eXtensible Markup Language*, lenguaje extensible de marcado), y el WML (*Wireless Markup Language*, lenguaje de marcado inalámbrico), el cual es un lenguaje de marcado basado en XML, diseñado para especificar el contenido y las interfaces de usuario de dispositivos inalámbricos de banda estrecha, tales como buscapersonas y teléfonos móviles, se consideran dentro del ámbito de la presente invención. Asimismo, a pesar de que en el presente documento se expone principalmente la norma HTML 3.2, se puede considerar cualquier versión de HTML dentro del ámbito de la presente invención.

Las comunicaciones entre el cliente 100 y el servidor web 116 pueden ser dirigidas usando una secuencia de solicitudes de HTTP 114 y respuestas de HTTP 112. A pesar de que el HTTP es descrito con referencia a una realización, otros protocolos de transporte, incluyendo el S-HTTP sin limitarse al mismo, se consideran dentro del ámbito de la presente invención. En el servidor web 116, un módulo de canalización de HTTP 118 recibe una solicitud de HTTP 114, resuelve el URL, e invoca un manipulador apropiado 120 para el procesamiento de la solicitud. En una realización de la presente invención, se prevé una pluralidad de manipuladores 120 para manejar diferentes tipos de recursos en el servidor web 116.

Por ejemplo, si el URL especifica un recurso de contenido estático 122, tal como un archivo de HTML, un manipulador 120 accede al recurso de contenido estático 122 y devuelve el recurso de contenido estático 122 a través de la canalización de HTTP 118 para la comunicación con el cliente 100 en una respuesta de HTTP 112. De forma alternativa, en una realización de la presente invención, si el URL especifica un recurso de contenido dinámico 124, tal como un recurso de ASP+, un manipulador 120 accede al recurso de contenido dinámico 124, procesa los contenidos del recurso de contenido dinámico 124, y genera el código de HTML resultante para la página 104 web. En una realización de la presente invención, el código de HTML resultante incluye el código de HTML 3.2 convencional. Por lo general, un recurso de contenido dinámico es una memoria de datos de declaraciones de lado de servidor (por ejemplo, un recurso de ASP+) que puede usarse para generar de forma dinámica el código de lenguaje de creación que describe una página web que se va a representar en un cliente. El código de HTML para la página web es pasado entonces a través de la canalización de HTTP 118 para la comunicación al cliente 100 en una respuesta de HTTP 112.

Durante su procesamiento, un manipulador 120 también puede acceder a bibliotecas de código de terceras partes o previamente desarrollado para simplificar el esfuerzo de desarrollo. Una biblioteca de ese tipo es una biblioteca de control de clases de lado de servidor 126, desde la cual el manipulador 120 puede instanciar objetos de control de lado de servidor para procesar elementos de interfaz de usuario y generar los datos de HTML resultantes para la representación de una página web. En una realización de la presente invención, uno o más objetos de control de lado de servidor establecen se encuentran en correspondencia con uno o más elementos de interfaz de usuario,

visibles u ocultos, en la página web que se describe en el recurso de contenido dinámico 124.

Una segunda biblioteca, por el contrario, es una biblioteca de clases de control de lado de cliente 128, tal como una biblioteca que incluya componentes de “ACTIVEX” de Microsoft Corporation. Un control de “ACTIVEX” es un objeto de COM (*Component Object Model*, Modelo de Objetos de Componentes) que sigue ciertas normas en cómo interactúa con su cliente y otros componentes. Un control de “ACTIVEX” de lado de cliente, por ejemplo, es un componente basado en COM que puede ser descargado automáticamente a un cliente y ejecutado por un navegador web en el cliente. Los componentes ACTIVEX de lado de servidor (que no se muestran) son componentes basados en COM que pueden ser implementados en un servidor para realizar una variedad de funciones de lado de servidor, tales como la provisión de la funcionalidad de lado de servidor de una aplicación de búsqueda de cotización de acciones o componente de base de datos. Se puede encontrar un análisis más detallado de ACTIVEX en el documento “*Understanding ACTIVEX and OLE*”, de David Chappell, Microsoft Press, 1996.

A diferencia de los controles de “ACTIVEX”, un objeto de control de lado de servidor en una realización de la presente invención, siendo especificado en un recurso de contenido dinámico 124, se corresponde de manera lógica con un elemento de interfaz de usuario que es representado en el cliente. El objeto de control de lado de servidor también puede generar código de HTML válido que puede incluir, por ejemplo, una marca HTML y un localizador que haga referencia a un control de “ACTIVEX” de lado de cliente dado. Si el navegador ya tiene el código para el control de “ACTIVEX” de lado de cliente dentro de su sistema de almacenamiento, el navegador ejecuta el control de “ACTIVEX” dentro de la página web en el cliente. De lo contrario, el navegador descarga el código para el control de “ACTIVEX” del recurso especificado por el localizador y entonces ejecuta el control de “ACTIVEX” dentro de la página web en el cliente. Un objeto de control de lado de servidor en una realización de la presente invención también puede levantar eventos a un objeto “ACTIVEX” de lado de servidor que se usa para implementar una aplicación de búsqueda de cotizaciones en el servidor.

Un manipulador 120 también tiene acceso a uno o más componentes de servidor no de interfaz de usuario 130 que ejecutan en el servidor web 116 o en otro servidor web accesible. Se puede hacer referencia a un componente de servidor no de interfaz de usuario 130, tal como una aplicación de búsqueda de cotización de acciones o componente de base de datos, en o asociado a un recurso de contenido dinámico 124 que es procesado por un manipulador 120. Los eventos de lado de servidor levantados por los objetos de control declarados en el recurso de contenido dinámico 124 pueden ser procesados por el código de lado de servidor, el cual llama a procedimientos apropiados en el componente de servidor no de interfaz de usuario 130. Como resultado, el procesamiento previsto por los objetos de control de lado de servidor simplifica la programación del componente de servidor no de interfaz de usuario 130 mediante el encapsulamiento del procesamiento y la generación de los elementos de interfaz de usuario de una página web, lo cual permite al desarrollador del componente de servidor no de interfaz de usuario 130 concentrarse en la funcionalidad específica de la aplicación, más que en cuestiones de interfaz de usuario.

La figura 2 ilustra un diagrama de flujo de operaciones para procesar y generar elementos de interfaz de usuario de lado de cliente usando objetos de control de lado de servidor en una realización de la presente invención. En la operación 200, el cliente transmite una solicitud de HTTP al servidor. La solicitud de HTTP incluye un URL que especifica un recurso, tal como un recurso de ASP+. En la operación 202, el servidor recibe la solicitud de HTTP e invoca el manipulador apropiado para procesar el recurso especificado. La solicitud de HTTP incluye una estructura de estado transportable que incluye una información de estado y, de forma opcional, una información jerárquica, que está asociada con uno o más objetos de control de lado de servidor, a pesar de que la primera solicitud de http para una página dada por lo general no incluye una estructura de estado transportable debido a que no ha tenido lugar cambio de estado alguno en el servidor para la página dada. Además, la estructura de estado transportable puede incluir una información de tipo de datos de propiedad, y un código de integridad para ayudar al servidor en la validación de la información de estado (es decir, verificar que la información de estado no estuviera corrompida en el cliente). La operación 204 genera una jerarquía de objetos de control de lado de servidor basándose en los contenidos del archivo de contenido dinámico especificado (por ejemplo, el recurso de ASP+). El recurso de ASP+ se lee en la operación 203. La operación 205 carga una información de estado que se recibe en la estructura de estado transportable en los objetos de control de lado de servidor apropiados en la jerarquía para restaurar los objetos de control a su estado previo.

En la operación 206, los objetos de control de lado de servidor de la jerarquía de objetos de control llevan a cabo una o más de las siguientes operaciones: el manejo de eventos de devolución, el manejo de datos de devolución, la gestión del estado, y el enlace de datos. Los eventos y datos de devolución (en conjunto “entrada de devolución”) de los elementos de interfaz de usuario están comunicados desde el cliente al servidor para el procesamiento. Un evento de devolución, por ejemplo, puede incluir sin limitación un evento de “clic de ratón” desde un elemento de botón de lado de cliente o un evento de “cambio de datos” desde un elemento de cuadro de texto de lado de cliente que está comunicado con el servidor. Los datos de devolución, por ejemplo, pueden incluir sin limitación texto introducido por un usuario en un elemento de cuadro de texto o un índice de un elemento seleccionado de un cuadro desplegable.

La operación 209 guarda los valores de propiedad, la información de tipo asociada y la información jerárquica en una estructura de estado transportable para la transmisión al cliente. En la operación 208, se llama a cada objeto de control de lado de servidor en la jerarquía para generar (o sintetizar) datos de lenguaje de creación, tales como

código de HTML, para la representación de elementos de interfaz de usuario de lado de cliente en la página web. Observe que, a pesar de que el término “sintetizar” puede ser usado para describir la operación de representar gráficos en una interfaz de usuario, el término “sintetizar” también es usado en el presente documento para describir la operación de generar datos de lenguaje de creación que pueden ser interpretados por la aplicación de cliente, tal como un navegador, para la representación y la funcionalidad de lado de cliente. La operación de síntesis 208 también genera los datos de lenguaje de creación que representan la estructura de estado transportable. Se proporciona un análisis más detallado de la operación de procesamiento 206 y la operación de síntesis 208 en relación con la figura 6. En una realización, las llamadas a procedimientos render() en objetos de control individuales son realizadas usando una secuencia de recorrido de árbol. Eso es, una llamada al procedimiento render() de un objeto de página da como resultado un recorrido recursivo a través de objetos de control de lado de servidor apropiados en la jerarquía. También se pueden emplear procedimientos alternativos para llamar a los procedimientos render() para objetos de control apropiados, incluyendo una señalización de eventos o enfoque de registro de objetos. Los paréntesis designan la etiqueta “render()” como indicador de un procedimiento, comparada con un valor de datos.

En una realización de la presente invención, la creación real de los objetos de control de lado de servidor individuales puede ser diferida hasta que se acceda al objeto de control de lado de servidor (tal como cuando se maneja entrada de devolución, se carga un estado, se sintetiza código de HTML a partir del objeto de control, etc.) en las operaciones 206 o 208. Si nunca se accede a un objeto de control de lado de servidor para una solicitud dada, la creación de objetos de control deferida optimiza el procesamiento de servidor mediante la eliminación de una operación de creación de objetos innecesaria.

La operación 210 transmite los datos de lenguaje de creación (por ejemplo, código de HTML), incluyendo la estructura de datos transportable, al cliente en una respuesta de HTTP. En la operación 214, el cliente recibe el código de HTML, incluyendo la estructura de datos transportable, asociado a una nueva página web que se va a representar. La estructura de datos transportable se puede almacenar en el cliente para su devolución al servidor en una solicitud de HTTP posterior. En la operación 216, el sistema de cliente incorpora (por ejemplo, dispositivos de representación) los elementos de interfaz de usuario de la nueva página de acuerdo con el código de HTML recibido a partir de la respuesta de HTTP. Debería entenderse, no obstante, que la incorporación de un elemento de interfaz de usuario puede incluir operaciones que no sean de representación, tales como la provisión de salida táctil o audio, la lectura y escritura en la memoria, el control de la operación de secuencias de comandos, etc. En la operación 212, se termina la jerarquía de objetos de control de lado de servidor. En una realización de la presente invención, los objetos de control de lado de servidor en la jerarquía son creados en respuesta a una solicitud de HTTP que haga referencia a un recurso de ASP+ asociado, y destruidos después de la síntesis de datos de lenguaje de creación (por ejemplo, datos de HTML). En una realización alternativa, la operación 212 puede ser realizada después de la operación 208 y antes de la operación 210.

La figura 3 ilustra un ejemplo de módulos en un servidor web que se usan en una realización de la presente invención. El servidor web 300 recibe una solicitud de HTTP 302 en la canalización de HTTP 304. La canalización de HTTP 304 puede incluir diversos módulos, tales como módulos para el registro de estadísticas de página web, la autenticación del usuario, y la memoria caché de salida de páginas web. Cada solicitud de HTTP entrante 302 recibida por el servidor web 300 es procesada en último lugar por una instancia específica de una clase de IHTTPHandler (que se muestra como el manipulador 306). El prefijo IHTTP indica que la clase es una Interfaz de un manipulador de HTTP. El manipulador 306 resuelve la solicitud de URL e invoca una fábrica de manipuladores apropiada (por ejemplo, un módulo de fábrica de páginas 308).

En la figura 3, un módulo de fábrica de páginas 308 en relación con el recurso de ASP+ 310 es invocado para manejar la instanciación y configuración de los objetos que se en el recurso de ASP+ 310. En una realización, un recurso de ASP+ puede ser identificado mediante la designación de un sufijo particular (o una extensión de archivo tal como “.aspx”) con el recurso. Cuando una solicitud para un recurso de ASP+ dado es recibido en primer lugar por el módulo de fábrica de páginas 308, el módulo de fábrica de páginas busca el sistema de archivos para el archivo apropiado (por ejemplo, el archivo .aspx 310). El archivo puede contener texto (por ejemplo, datos de lenguaje de creación) u otro formato de datos (por ejemplo, datos en código bytecode o datos codificados) que el servidor puede interpretar o al que puede acceder más tarde para atender la solicitud. Si el archivo físico existe, el módulo de fábrica de páginas 308 abre el archivo y escribe por lectura el archivo en la memoria. Si no se puede encontrar el archivo, el módulo de fábrica de páginas 308 devuelve un mensaje de error apropiado “archivo no encontrado”.

Después de escribir por lectura el recurso de ASP+ 310 en la memoria, el módulo de fábrica de páginas 308 procesa el contenido del archivo para construir un modelo de datos de la página (por ejemplo, listas de bloques de secuencia de comandos, directivas, zonas de texto estático, objetos de control jerárquicos de lado de servidor, propiedades de control de lado de servidor, etc.). El modelo de datos es usado para generar un listado de origen una nueva clase de objetos, tal como una clase COM+ (“*Component Object Model*+”, Modelo de Objetos de Componentes+), que extiende la clase base de páginas. La clase base de páginas incluye el código que define la estructura, propiedades, y la funcionalidad de un objeto de página básico. El listado de origen es entonces compilado de manera dinámica a un lenguaje intermedio. Un lenguaje intermedio puede incluir código de lenguaje general o realizado según la costumbre, tal como un código COM+ IL, códigos bytecode Java, código Modula 3, código SmallTalk y código Visual Code. En una realización alternativa, se pueden omitir las operaciones de lenguaje intermedias, de tal modo que las

instrucciones nativas son generadas directamente a partir del listado de origen o el archivo del recurso (por ejemplo, el recurso de ASP+ 310). Se puede acceder a una biblioteca de clases de control 312 mediante el módulo de fábrica de páginas 308 para obtener clases predefinidas de control de lado de servidor que se usan en la generación de la jerarquía de objetos de control.

El módulo de fábrica de páginas 308 produce un objeto de página 314, el cual es un objeto de control de lado de servidor que se corresponde con la página 104 web de la figura 1. El objeto de página 314 y sus hijos (es decir, un objeto de cuadro de texto 318, un objeto de botón 320, y otro objeto de botón 322) comprenden un ejemplo de jerarquía de objetos 316 de control. También se considera de acuerdo con la presente invención otro ejemplo de objetos de control, incluyendo sin limitación objetos correspondientes a los controles HTML de la tabla 1, así como objetos de control habituales. El objeto de página 314 se corresponde de manera lógica con la página 104 web de la figura 1. El objeto de cuadro de texto 318 se corresponde con el cuadro de texto 106 de la figura 1. Asimismo, el objeto de botón 320 se corresponde con el botón de añadir 108 de la figura 1, y el objeto de botón 322 se corresponde con el botón de suprimir 110 de la figura 1. El objeto de página 314 está relacionado jerárquicamente con otros objetos de control en el servidor. En una realización, un objeto de página es un objeto de contenedor que contiene jerárquicamente sus objetos de control hijo. En una realización alternativa, se pueden emplear otras formas de relación jerárquica, incluyendo una relación de dependencia. En una jerarquía de objetos de control más compleja con múltiples niveles de hijos, un objeto hijo puede ser un objeto de contenedor para otros objetos hijo.

En la realización que se ilustra, los objetos de control en la jerarquía de objetos 316 de control son creados y ejecutados en el servidor 3000, y cada objeto de control de lado de servidor funciona como un “reflejo” de un elemento de interfaz de usuario correspondiente de cliente. Los objetos de control de lado de servidor de la presente realización también cooperan para manejar la entrada a partir de la solicitud de HTTP 302, para gestionar los estados de objetos de control de lado de servidor, para realizar el enlace de datos con bases de datos de lado de servidor, y para generar datos de lenguaje de creación (por ejemplo, código de HTML) que se usan para representar una página web resultante en un cliente. Los datos de lenguaje de creación resultantes son generados (por ejemplo, sintetizados) a partir de la jerarquía de objetos de control de lado de servidor 316 y transmitidos al cliente en una respuesta de HTTP 324. Por ejemplo, el código de HTML resultante puede plasmar cualquier construcción de HTML válida y puede hacer referencia a controles de tipo ACTIVEX, miniaplicaciones JAVA, secuencias de comandos, y otros recursos web cualesquiera que den como resultado elementos de interfaz de usuario de lado de cliente (por ejemplo, botones de control, cuadros de texto, etc.) cuando son procesados por un navegador.

En virtud de declaraciones hechas en el recurso de ASP+ 310, los objetos de control de lado de servidor pueden acceder a uno o más componentes de servidor no de interfaz de usuario 330 para proporcionar interacción entre el componente de servidor no de interfaz de usuario 330 y elementos de interfaz de usuario de lado de cliente. Por ejemplo, en respuesta a la entrada de devolución, los objetos de control de lado de servidor pueden levantar eventos de lado de servidor a los componentes de servidor no de interfaz de usuario registrados para esos eventos. De este modo, el componente de servidor no de interfaz de usuario 330 puede interactuar con el usuario a través de elementos de la interfaz del usuario sin programar el código requerido para representar y procesar estos elementos.

En resumen, una realización de la presente invención incluye objetos de control de lado de servidor que son creados y ejecutados en el servidor para generar código de HTML que es enviado a un cliente. El código de HTML puede, por ejemplo, representar cualquier construcción de HTML válida y puede hacer referencia a controles de tipo ACTIVEX, miniaplicaciones JAVA, secuencias de comandos, y cualesquiera otros recursos web para producir botones de interfaz de usuario y otros elementos de la interfaz en el cliente. Un usuario en el cliente puede interactuar con estos elementos de interfaz de usuario, los cuales se corresponden de manera lógica con los objetos de control de lado de servidor, y envían una solicitud de vuelta al servidor. Los objetos de control de lado de servidor son recreados en el servidor para procesar los datos, eventos, y otras características de los elementos de interfaz de usuario para generar la siguiente ronda de código HTLM que ha de transmitirse en una respuesta al cliente.

Haciendo referencia a la figura 4, un ejemplo de sistema informático para realizaciones de la invención incluye un dispositivo informático de aplicaciones generales en la forma de un sistema informático 400, incluyendo una unidad de procesamiento 402, una memoria de sistema 404 y un bus de sistema 406 que une diversos componentes del sistema incluida la memoria de sistema 404 a la unidad de procesamiento 400. El bus de sistema 406 puede ser cualquiera de distintos tipos de estructuras de bus incluido un bus de memoria o controlador de memoria, un bus periférico y un bus local usando cualquiera de una variedad de arquitecturas de bus. La memoria de sistema incluye memoria de solo lectura (ROM, *read only memory*) 408 y memoria de acceso aleatorio (RAM, *random access memory*) 410. Un sistema básico de entrada / salida 412 (BIOS, *Basic Input / Output System*), el cual contiene rutinas básicas que ayudan a transferir información entre elementos dentro del sistema informático 400, es almacenado en la ROM 408.

El sistema informático 400 incluye además una unidad de disco duro 412 para leer desde ella y escribir en un disco duro, una unidad de disco magnético 414 para leer desde ella o escribir en un disco magnético extraíble 416, y una unidad de disco óptico 418 para leer desde ella o escribir en un disco óptico extraíble 419 tal como un CD ROM, DVD, u otros medios ópticos. La unidad de disco duro 412, la unidad de disco magnético 414, y la unidad de disco óptico 418 son conectadas al bus de sistema 406 mediante una interfaz de la unidad de disco duro 420, una interfaz de la unidad de disco magnético 422, y una interfaz de la unidad óptica 424, respectivamente. Las unidades y sus

medios asociados legibles por ordenador proporcionan almacenamiento no volátil de instrucciones legibles por ordenador, estructuras de datos, programas, y otros datos para la sistema informático 400.

A pesar de que el entorno del ejemplo que se describe en el presente documento emplea un disco duro, un disco magnético extraíble 416, y un disco óptico extraíble 419, se pueden usar otros tipos de medios legibles por ordenador que pueden almacenar datos en el sistema del ejemplo. Ejemplos de estos otros tipos de medios legibles por ordenador que pueden ser usados en el ejemplo de entorno operativo incluyen cintas magnéticas, tarjetas de memoria flash, discos digitales de vídeo, cartuchos de Bernoulli, memorias de acceso aleatorio (RAM) y memorias de solo lectura (ROM).

Un número de módulos de programa puede ser almacenado en el disco duro, el disco magnético 416, el disco óptico 419, la ROM 408 o la RAM 410, incluyendo un sistema operativo 426, uno o más programas de aplicación 428, otros módulos de programa 430 y unos datos de programa 432. Un usuario puede introducir órdenes e información en el sistema informático 400 a través de dispositivos de entrada tales como un teclado 434 y un ratón 436 u otro dispositivo de puntero. Ejemplos de otros dispositivos de entrada pueden incluir un micrófono, una palanca de mando, un tablero de juego, un plato parabólico, y un escáner. Éstos y otros dispositivos de entrada están conectados frecuentemente a la unidad de procesamiento 402 a través de una interfaz de puerto serie 440 que está acoplada al bus de sistema 406. No obstante, estos dispositivos de entrada también pueden estar conectados mediante otras interfaces, tales como un puerto paralelo, un puerto de juegos o un bus serie universal (USB, *Universal Serial Bus*). Un monitor 442 u otro tipo de dispositivo de representación también está conectado al bus de sistema 406 por medio de una interfaz, tal como un adaptador de vídeo 444. Además del monitor 442, los sistemas informáticos incluyen normalmente otros dispositivos periféricos de salida (que no se muestran), tales como altavoces e impresoras.

El sistema informático 400 puede operar en un entorno en red usando conexiones lógicas a uno o más ordenadores remotos, tales como un ordenador remoto 446. El ordenador remoto 446 puede ser un sistema informático, un servidor, un router, un ordenador personal de red, un dispositivo entre iguales u otro nodo de red común, y normalmente incluye muchos de, o todos, los elementos que se han descrito en lo que antecede relativos al sistema informático 400. Las conexiones de la red incluyen una red de área local (LAN, *local area network*) 448 y una red de área extensa (WAN, *wide area network*) 450. Tales entornos de trabajo en red son habituales en oficinas, redes informáticas a nivel empresarial, intranets e Internet.

Cuando se usa en un entorno de trabajo en red LAN, el sistema informático 400 está conectado a la red local 448 a través de una interfaz de red o adaptador 452. Cuando se usa en un entorno de trabajo en red WAN, el sistema informático 400 incluye normalmente un módem 454 u otros medios para establecer comunicaciones por la red de área extensa 450, tales como Internet. El módem 454, el cual puede ser interno o externo, está conectado al bus de sistema 406 por medio de la interfaz de puerto serie 440. En un entorno de red, los módulos de programa representados relativos al sistema informático 400, o partes del mismo, pueden estar almacenados en el dispositivo remoto de almacenamiento de memoria. Se considerará que las conexiones de la red que se muestran son ejemplos, y que se pueden usar otros medios de establecimiento de un enlace de comunicación entre los ordenadores.

En una realización de la presente invención, el ordenador 400 representa un servidor web, en el que la CPU 402 ejecuta un módulo de fábrica de páginas en un recurso de ASP+ almacenado en al menos uno de medios de almacenamiento 416, 412, 414, 418, 419, o la memoria 404. Las respuestas y las solicitudes de HTTP son comunicadas por la LAN, 448, que está conectada al ordenador de cliente 446.

La figura 5 ilustra un diagrama de flujo del procedimiento que representa el procesamiento de lado de servidor de un objeto de página y otros objetos de control en una realización de la presente invención. En la operación 500, se llama a un constructor de objetos de página mediante el módulo de fábrica de páginas 308 (véase la figura 3). Como resultado, un objeto de página (véase, por ejemplo, el objeto de página 314 en la figura 3) se crea para que funcione como un “reflejo” del elemento de interfaz de usuario de página web en el cliente. En la operación 502, el módulo de fábrica de páginas llama a la función del elemento ProcessRequest() del objeto de página, el cual inicia las operaciones por fases para el procesamiento de la solicitud de HTTP recibida de un cliente. En una primera fase de una realización de la presente invención, una operación de creación de lado de servidor (que no se muestra) puede crear los objetos de control de lado de servidor descendente contenidos en la jerarquía de objetos de control del objeto de página, es decir, se llama a los constructores para objetos de control hijo de manera recurrente para crear los objetos de control durante la duración del procesamiento de la solicitud de HTTP.

En una realización alternativa, no obstante, la creación de objetos de control hijo es deferida hasta que el objeto de control es requerido para una etapa de procesamiento dada (por ejemplo, el manejo de un evento de devolución, el manejo de datos de devolución, la carga o guardado de un estado de vista, la resolución de enlace de datos, o la síntesis de código de HTML para el elemento de interfaz de usuario correspondiente). La última realización, la cual se dice que implementa “la creación deferida de objetos de control”, es una optimización que puede aliviar la CPU y utilización de memoria innecesarias. Por ejemplo, un evento de entrada de usuario recibido de cliente puede tener como resultado la creación de una página web completamente diferente. En este caso, no es necesario instanciar una jerarquía completa de objetos de control de la página anterior solo para procesar un evento que tiene como

resultado, de forma inmediata, la terminación de la jerarquía de objetos de control y la instanciación de una jerarquía de objetos de control nueva y diferente para una nueva página.

En respuesta a la llamada de servidor al procedimiento ProcessRequest del objeto de página, el objeto de página y objetos de control descendente individuales pueden ejecutar las operaciones 504 a 520, dependiendo en parte de los datos de una solicitud de HTTP dada. En una realización de la presente invención, las operaciones 504 a 520 son realizadas para cada objeto individual en el orden que se ilustra en la figura 5; no obstante, una operación dada para un objeto puede producirse fuera de orden o no con respecto a una operación dada de otro objeto, dependiendo de la solicitud de HTTP. Por ejemplo, un primer objeto puede realizar su operación de inicialización 504 y su operación de carga 506, y comenzar la operación de procesamiento de datos de devolución 508, antes de que un objeto de control descendente realice su propia operación de inicialización 504 y la operación de carga 506 en virtud de la creación deferida de objetos de control. El orden de procesamiento de operaciones mediante el objeto de página y los objetos de control descendente depende de diversos factores, incluyendo sin limitarse a los mismos, de la naturaleza de los datos en la solicitud de HTTP, de la configuración de la jerarquía de objetos de control, del estado actual de los objetos de control y de si está implementada la creación diferida de objetos de control.

La operación de inicialización 504 inicializa un objeto de control después de que éste sea creado mediante la ejecución de cualquier código de lado de servidor asociado a la inicialización en el archivo de contenido dinámico. De esta forma, cada objeto de control de lado de servidor puede ser adaptado con la funcionalidad de lado de servidor específica que es declarada en el archivo de contenido dinámico. En una realización de la presente invención, el código de contenido dinámico destinado a adaptar o extender las clases base de control de página tal como se declara mediante el desarrollador de páginas en el recurso de ASP+ en el servidor. Cuando el recurso de ASP+ es compilado, el código declarado es incluido en el código de inicialización apropiado (por ejemplo, los procedimientos Init() (inicialización) del objeto de página y los objetos de control descendente). La operación de inicialización 504 ejecuta este código para adaptar o extender la clase base de página y las clases base para objetos de control descendente.

En una realización de la presente invención, la gestión del estado de los objetos de control de lado de servidor es soportada en una operación de carga 506 y una operación de guardado 516, las cuales usan una estructura de estado transportable para alojar el modelo sin estado para sistemas cliente-servidor mediante la restauración de objetos de control de lado de servidor para sus estados previos. En una realización, el estado es comunicado a y desde servidor en un o más campos HTML ocultos de un par solicitud / respuesta de HTTP, a pesar de que otras estructuras de estado transportables se consideran dentro del ámbito de la presente invención, incluyendo *cookies* y campos visibles.

En una secuencia dada de solicitudes y respuestas relacionadas con la página actual entre un cliente y un servidor, los estados de uno o más objetos de control son registrados en una estructura de estado transportable por la operación de guardado 516 después del procesamiento de una solicitud previa. En una realización de la presente invención, también se incluye información adicional del estado en la estructura de estado transportable, incluyendo información jerárquica o identificadores de objetos de control para permitir al servidor asociarse a un estado dado con el objeto de control apropiado. En una solicitud de HTTP posterior, la información de estado es devuelta al servidor en la estructura de estado transportable. El servidor extrae la información de estado de la estructura de estado transportable recibida y carga los datos del estado en los objetos de control apropiados dentro de la jerarquía de objetos de control para restaurar cada objeto de control a su estado como existía antes de una respuesta de HTTP previa. Después de que la solicitud actual sea procesada, los estados de uno o más objetos de control de lado de servidor son registrados otra vez en la estructura de estado transportable por la operación de guardado 516, y la estructura de estado transportable es devuelta al cliente en la siguiente respuesta de HTTP.

Como resultado de la operación de carga 506, cada objeto de control de lado de servidor es colocado en un estado concordante con su estado antes de una respuesta de HTTP anterior. Por ejemplo, si un objeto de control de cuadro de texto incluye un valor de propiedad igual a “JDoe” antes de una respuesta de HTTP anterior, la operación de carga 506 restaura el mismo objeto de control a su estado anterior, en parte cargando la secuencia de texto “JDoe” en el valor de propiedad. Además, si el estado de un objeto dado es almacenado y restaurado es configurable.

Resumiendo una realización de la presente invención, el estado de uno o más objetos de control de lado de servidor es “guardado” después del procesamiento. La información de estado guardada es transmitida al cliente en una respuesta. El cliente devuelve la información de estado guardada al servidor en una respuesta siguiente. El servidor carga la información de estado en una jerarquía de objetos de control de lado de servidor recién instanciados, tales que el estado de la jerarquía es restaurado a su estado previo.

Una realización alternativa puede mantener la información de estado en el servidor o en alguna otra ubicación web accesible por el servidor durante el viaje de ida y vuelta del servidor al cliente, y luego de vuelta al servidor. Después de que la solicitud de cliente sea recibida por el servidor, esta información de estado puede ser recuperada por el servidor y cargada en el objeto u objetos de control de lado de servidor apropiados en la jerarquía de objetos de control.

En la operación 508, se procesan datos de devolución recibidos de la solicitud de HTTP. Se pueden incluir datos de

devolución en la carga útil de la solicitud de HTTP en pares de valores de tecla, en una representación jerárquica (por ejemplo, XML), o en otras representaciones de datos, tales como RDF ("Marco de Descripción de Recursos"). La operación 508 analiza la carga útil para identificar un identificador único de un objeto de control de lado de servidor. Si se encuentra el identificador (por ejemplo "page1:text1") y el objeto de control de lado de servidor identificado existe en la jerarquía de objetos de control, los datos de devolución correspondientes son pasados al objeto de control. Por ejemplo, haciendo referencia a la figura 1, un identificador único asociado con el cuadro de texto 106 y el texto "JDoe" son comunicados en la carga útil de la solicitud de HTTP 114 al servidor web 116. La operación 508 analiza la carga útil de la solicitud de HTTP y obtiene el identificador único del cuadro de texto 106 y su valor asociado (es decir, "JDoe"). La operación 508 resuelve entonces el identificador único del cuadro de texto 106 para identificar el objeto de control de lado de servidor correspondiente y pasa el valor "JDoe" al objeto para el procesamiento.

Tal como se ha analizado con respecto a la operación de carga 506, los valores de propiedad de objetos de control de lado de servidor pueden ser restaurados a sus estados previos. En respuesta a la recepción de datos de devolución, el objeto de control de lado de servidor determina si el valor de devolución que se ha pasado da lugar a un cambio con respecto al valor anterior de la propiedad correspondiente. Si es así, el cambio es registrado en una lista de cambios para indicar un cambio de datos para el objeto de control asociado. Después de que todos los datos de devolución hayan sido procesados dentro de la jerarquía de objetos de control, se puede hacer una llamada a un procedimiento de objetos de control para levantar uno o más eventos cambiados de datos de devolución a uno o más componentes de servidor no de interfaz de usuario, tales como una aplicación de búsqueda de cotización de acciones que se ejecuta en el servidor. Un ejemplo de un evento cambiado de datos de devolución es un evento que indique que esos datos de devolución han dado lugar a que cambie una propiedad de un objeto de control de lado de servidor. En un ejemplo de realización, un evento de tales características puede ser enviado a una cola de eventos provista por el sistema, de tal modo que se pueda invocar un componente de servidor que se registra para procesar el evento. De esta forma, un componente de servidor no de interfaz de usuario de lado de servidor puede responder a eventos activados por un cambio en los datos de un objeto de control de lado de servidor. También se consideran dentro del ámbito de la presente invención procedimientos alternativos de implementación de eventos, incluyendo el uso de colas de eventos provistas por la aplicación, sondeo, e interrupciones del procesamiento.

En la operación 510, se manejan los eventos de devolución. Los eventos de devolución se comunican en la carga útil de la solicitud de HTTP. La operación 510 analiza un objetivo de evento especificado (por ejemplo, "_EVENTTARGET" etiquetado en una realización de la presente invención) que identifica el objeto de control de lado de servidor al cual se dirige el evento. Asimismo, la operación 510 analiza los argumentos del evento localizados, si los hubiera, y proporciona el argumento del evento (por ejemplo, "_EVENTARGUMENT" etiquetado en una realización de la presente invención) al objeto de control de lado de servidor especificado. El objeto de control levanta sus eventos para el procesamiento mediante el componente de servidor no de interfaz de usuario (por ejemplo, unas aplicaciones de búsqueda de cotización de acciones de lado de servidor) asociado al recurso de contenido dinámico.

La operación 512 resuelve el enlace de datos entre los objetos de control de lado de servidor y una o más bases de datos accesibles por el servidor. En una realización de la presente invención, las propiedades de objeto de control de lado de servidor se pueden asociar (o enlazar por datos) a propiedades de un contenedor de enlace de datos padre, tales como una tabla en una base de datos de la aplicación de lado de servidor. Durante la operación de enlace de datos 612, la estructura de la página puede actualizar una propiedad de objetos de control enlazada por datos con el valor de la propiedad del contenedor de enlace de datos padre correspondiente. De esta forma, los elementos de interfaz de usuario en la página web de la siguiente respuesta reflejan con exactitud unos valores de propiedad actualizados, puesto que las propiedades de objetos de control a las cuales corresponden los elementos de interfaz de usuario han sido actualizadas automáticamente durante la operación de enlace de datos 512. Asimismo, las propiedades de objeto de control también pueden ser actualizadas a los campos de contenedor de enlace de datos padre, actualizando de esta forma una base de datos de la aplicación de lado de servidor con entrada de devolución de un objeto de control de lado de servidor.

La operación 514 realiza diversas operaciones de actualización que pueden ser ejecutadas antes de que se guarde el estado del objeto de control y se sintetice la salida. La operación 516 solicita información de estado (es decir, estado de vista) de uno o más objetos de control en la jerarquía de objetos de control y almacena la información de estado para la inserción en una estructura del estado transportable que está comunicada con el cliente en la carga útil de la respuesta de HTTP. Por ejemplo, un objeto de control de "retícula" puede guardar una página de índice actual de una lista de valores, de manera que el objeto de control de "retícula" pueda ser restaurado a su estado después de una solicitud de HTTP posterior (es decir, en la operación 506). Tal como se describe en lo que antecede, la información de estado de vista representa el estado de la jerarquía de objetos de control antes de cualquier acción posterior de cliente (por ejemplo, antes de que se envíe al cliente la respuesta de HTTP). Cuando la información de estado de vista es devuelta, será usada para colocar la jerarquía de objetos de control en ese estado anterior antes del procesamiento de cualquier entrada de devolución de cliente o enlace de datos.

La operación de síntesis 518 genera la salida de lenguaje de creación apropiada (por ejemplo, datos de HTML) para la comunicación al cliente en una respuesta de HTTP. La síntesis se efectúa a través de un recorrido de árbol jerárquico descendente de todos los objetos de control de lado de servidor y el código de síntesis incrustado. La

operación 520 realiza cualquier trabajo de limpieza final (por ejemplo, el cierre de archivos o de conexiones de bases de datos) antes de que se finalice la jerarquía de objetos de control. El procesamiento vuelve entonces a la operación 502 y avanza a la operación 522 en la que el objeto de página se termina llamando a su destructor.

La figura 6 ilustra una porción a modo de ejemplo de un archivo de contenido dinámico (por ejemplo, un recurso de ASP+) en una realización de la presente invención. La línea 1 del recurso de ASP+ 600 es una etiqueta de inicio de un archivo de HTML y se declara en el recurso de ASP+ 600 como un literal. Un literal se corresponde con un objeto de control de literal de lado de servidor en la jerarquía de objetos de control de lado de servidor. Al objeto de control de literal que se corresponde con la línea 1 se le da un índice de “0” debido a que este es el primer objeto de control que resulta de una declaración en el recurso de ASP+ 600. En el momento de la síntesis, el objeto de control de literal meramente genera el texto “<html>” y una nueva línea para la inclusión en la respuesta de HTTP. Las líneas 2 a 12 del recurso de ASP+ 600 representan un bloque de declaración de código, que se ejecuta en el servidor (es decir, tal como se indica mediante el atributo “runat = server” en la línea 2). En el recurso de ASP+ 600 a modo de ejemplo, el bloque de declaración de código no da como resultado la instanciación de un objeto de control de lado de servidor en la jerarquía de objetos de control. En su lugar, el bloque de declaración de código da como resultado que un código de lado de servidor se “cablee” o se asocie con el objeto de control de lado de servidor que se declara en la línea 15. La línea 13 del recurso de ASP+ 600 es una etiqueta de inicio del cuerpo del archivo de HTML y se declara en el recurso de ASP+ 600 como un literal para dar como resultado un objeto de control de literal que tiene un índice que es igual a “1”.

La declaración en la línea 14 declara un objeto de control de formulario de lado de servidor (índice = “2”) que se va a instanciar en la jerarquía de objetos de control. Las etiquetas en la línea 17 y 18 son unas etiquetas de cierre que se van a representar mediante un objeto de control de literal de lado de servidor (índice = “3”) en la jerarquía de objetos de control. La etiqueta de cierre en la línea 16 cierra las declaraciones de <form> (formulario) que se corresponden con el objeto de control de formulario.

En un nivel posterior de jerarquía, la declaración en la línea 15 declara un objeto de control de etiqueta de lado de servidor (índice = “1”) que usa la etiqueta de control de HTML “span” (extensión) y que tiene el atributo de identificador (“id”) que es igual a “Message” (mensaje). Los índices “0” y “2” de este nivel de jerarquía se asignan, de forma respectiva, a un objeto de control de literal (índice = “0”) para un literal de espacio en blanco precedente (por ejemplo, tabulaciones, nuevas líneas, espacios, etc.) y un objeto de control de literal (índice = “2”) de un literal de espacio en blanco posterior.

La figura 7 ilustra un código resultante que es generado por uno o más objetos de control de lado de servidor en respuesta a la porción a modo de ejemplo del archivo de contenido dinámico de la figura 6. Solo las secciones de form (formulario), de gestión de estados y de span (extensión) se ilustran en una porción de código de HTML 700 para facilitar este análisis. Otras etiquetas, tales como las etiquetas de <html> y de <body> (cuerpo), no están incluidas en la porción de código de HTML 700, a pesar de que el código de HTML para estas etiquetas se incluiría en un conjunto de código de HTML completo que resulta del recurso de ASP+ de la figura 6.

Las líneas 1 y 5 son unas etiquetas de inicio y de fin de la sección de formulario del código de HTML y son generadas por los objetos de control de formulario de lado de servidor que se declaran en las líneas 14 a 16 de la figura 6. La línea 4 incluye el código de HTML para una etiqueta que representa la fecha de la operación de Última Devolución (es decir, “Última Devolución: 6 / 5 / 99”), que se declaró en la línea 15 de la figura 6.

Los campos ocultos en las líneas 2 y 3 representan una realización de una estructura de estado transportable de acuerdo con la presente invención. En la línea 2, la información de estado de los objetos de control de lado de servidor que se declaran en el recurso de ASP+ 600 se registra en el campo oculto que se denomina “_VIEWSTATE”. El valor del campo de _VIEWSTATE es una cadena de texto que representa los valores de estado, los tipos de propiedad y la información jerárquica de los objetos de control en la jerarquía de objetos de control. En el campo de _VIEWSTATE también se incluye una información jerárquica para permitir que la operación de carga recorra la jerarquía y cargue un valor de estado dado en una propiedad de un objeto de control apropiado en la jerarquía.

En una realización de la presente invención, un cliente interacciona con un servidor web usando una secuencia de pares de solicitud / respuesta de HTTP. Entre una respuesta al cliente y la siguiente solicitud que es recibida por el servidor, puede que el servidor no mantenga el estado de los objetos de control de lado de servidor que están asociados con un recurso de ASP+ dado o una conexión dada con el cliente. En su lugar, la información de estado para una jerarquía de objetos de control dada se envía al cliente en una estructura de estado transportable (por ejemplo, en la respuesta de HTTP), y se devuelve al servidor en la estructura de estado transportable (por ejemplo, en la siguiente solicitud de HTTP).

La línea 3 representa un campo oculto que se denomina “_VIEWSTATEMAC” que tiene un valor de “434333433”. El _VIEWSTATEMAC es un código de integridad en la estructura de estado transportable que es usado por la operación de carga para verificar que el valor de _VIEWSTATE no estuviera corrompido en el cliente. El valor de _VIEWSTATEMAC se calcula inicialmente en el servidor a partir de los contenidos del valor de _VIEWSTATE y se comunica al cliente en una respuesta. Cuando el cliente devuelve la estructura de estado transportable, el servidor

web calcula un nuevo código de integridad a partir de los contenidos del valor de _VIEWSTATE recibido. Si el código de integridad recibido y el código recién calculado son iguales, entonces el servidor web supone que el valor de _VIEWSTATE recibido es válido o, por lo demás, que no está corrompido (es decir, es el mismo que el valor de _VIEWSTATE que se envió previamente al cliente). En una realización de la presente invención, un algoritmo MD5 se usa para calcular el código de integridad. MD5 es un algoritmo creado en 1991 por el profesor Ronald Rivest para su uso en la creación de firmas digitales. MD5 es una función de hash unidireccional, lo que quiere decir que esta toma un mensaje y lo convierte en una cadena fija de dígitos, que también se denomina resumen de mensaje. Tales procedimientos pueden incluir el uso de una clave secreta para desalentar la manipulación indebida por parte de individuos o programas no autorizados. Otras realizaciones, no obstante, pueden incluir unas técnicas de codificación de integridad alternativas, incluyendo generar un código basándose en el algoritmo MD4 o cualquier otro algoritmo de hash, o la longitud del campo de _VIEWSTATE.

En una realización de la presente invención, la estructura del campo de _VIEWSTATE de la línea 2 representa un anidamiento jerárquico de objetos de control y sus estados (por ejemplo, unos valores de estado y sus tipos de datos de propiedad asociados). Las etiquetas que se usan en un campo de _VIEWSTATE a modo de ejemplo se describen en la tabla 1. Las etiquetas que incluyen un carácter "/" representan unas etiquetas de cierre.

Tabla 1

Etiqueta	**Descripción**
<s>, </s>	valor de cadena
<a*x*>, 	matriz que tiene *x* elementos
<i>	valor entero
<h*x*>, </H>	tabla de hash que tiene *x* entradas de tabla
<K>, </K>	clave de tabla de hash
<V>, </V>	valor de tabla de hash
,	valor Booleano
<d>, </d>	valor de fecha/hora
<c>, </c>	valor de moneda
<A>, </A>	lista de matriz
<n>, </n>	valor nulo

Se ha de entender que en el presente documento se divulga una realización de codificación a modo de ejemplo, a pesar de que, en realizaciones alternativas, se pueden emplear otros procedimientos de codificación. Por ejemplo, una información de jerarquía, unos valores de estado y sus tipos de datos de propiedad asociados se pueden designar usando unos identificadores jerárquicos únicos para cada objeto de control de lado de servidor, o se puede emplear un formato de datos relacionado con XML para representar los datos en una estructura de estado transportable. Además, se pueden usar técnicas de cifrado y de compresión conocidas para proporcionar seguridad y para reducir el tamaño de la estructura de estado transportable.

La figura 8 ilustra la jerarquía de objetos de control que se corresponde con el recurso de ASP+ de la figura 6 y el valor de campo de _VIEWSTATE de la figura 7 en una realización de la presente invención. Un bloque de comprobación de archivo 802, a pesar de que no es un componente de objeto de control de la jerarquía de objetos de control, se ilustra como un nivel de arriba de jerarquía basándose en la estructura del campo de _VIEWSTATE, tal como se muestra en la figura 7. Una descripción más detallada del bloque de comprobación de archivo 802 se proporciona con respecto a la figura 9. Como alternativa, se contempla dentro del ámbito de la presente invención la omisión del bloque de comprobación de archivo 802 o la fusión del mismo en el nivel de jerarquía que se corresponde con un objeto de control de página 804.

El objeto de control de página 804 se corresponde con un objeto de control especial que se instancia como el nivel superior de la jerarquía de objetos de control y que se corresponde con la propia página web resultante. En una realización de la presente invención, no se requiere declaración (es decir, una declaración que tiene el atributo "runat = server") específica alguna para dar lugar a una instanciación del objeto de página 804.

En el siguiente nivel de jerarquía, los objetos de control 806, 808, 810 y 812 se instancian como objetos de control hijos que están contenidos por o que están relacionados de forma jerárquica con el objeto de página 804. A cada objeto de control en un nivel de jerarquía se le asigna un índice de base cero basándose en su orden de arriba abajo en el recurso de ASP+. Por ejemplo, el objeto de control de literal 806 se corresponde con el texto "<html>" y los espacios en blanco asociados en la línea 1 del recurso de ASP+ 600 en la figura 6. Por consiguiente, al objeto de control de literal 806 se le asigna un índice que es igual a "0". En la realización ilustrada de la presente invención, el bloque de declaración de código que se extiende a partir de las líneas 2 - 12 no da como resultado la instanciación de un objeto de control correspondiente. El objeto de control de literal 808 se corresponde con el texto "<body>" y el espacio en blanco asociado en la línea 13 del recurso de ASP+ 600 de la figura 6. Por consiguiente, al objeto de control de literal 808 se le asigna un índice que es igual a "1". El objeto de control de formulario 810 se corresponde

con la declaración de form (formulario) en las líneas 14 a 16 del recurso de ASP+ 600 de la figura 6. Por consiguiente, al objeto de control de formulario 810 se le asigna un índice que es igual a “2”. El objeto de control de literal 812 se corresponde con las etiquetas de cierre y el espacio en blanco asociado en las líneas 17 y 18 del recurso de ASP+ 600 de la figura 6. Por consiguiente, al objeto de control de literal 812 se le asigna un índice que es igual a “3”.

En el siguiente nivel de jerarquía, los objetos de control 814, 816 y 818 se instancian como objetos de control hijos que están contenidos por o que están relacionados de forma jerárquica con el objeto de control de formulario 810. El objeto de control de literal 814 se corresponde con un espacio en blanco que precede a la declaración de span (extensión) en la línea 15 del recurso de ASP+ 600 de la figura 6. Por consiguiente, al objeto de control de literal 814 se le asigna un índice de “0”, debido a que este es el primer texto de tipo literal (incluso si es un espacio en blanco) que se encuentra en asociación con este nivel de jerarquía en el recurso de ASP+. De forma similar, el objeto de control de literal 818 se corresponde con un espacio en blanco que sigue a la declaración de span (extensión) de la línea 15 en el recurso de ASP+ 600 de la figura 6. Por consiguiente, al objeto de control de literal 818 se le asigna un índice de “2”. El objeto de control de etiqueta 816 se corresponde con la declaración en la línea 15 del recurso de ASP+ 600 en la figura 6. Por consiguiente, al objeto de control de etiqueta 816 se le asigna un índice que es igual a “1”.

La figura 9 ilustra una versión anidada del valor de campo de _VIEWSTATE de la figura 7. La versión anidada se muestra en el presente documento para facilitar el análisis de la naturaleza jerárquica del campo de _VIEWSTATE. La línea 1 de la versión anidada 900 se corresponde con el comienzo del bloque de comprobación de archivo anidado 802 de la figura 8. La línea 1 especifica una matriz de dos elementos, que comprende una cadena en la línea 2 como el primer elemento y una matriz de dos elementos de la línea 3 como el segundo elemento. La línea 2 especifica un valor de cadena que representa un código de comprobación de archivo. En una realización de la presente invención, el código de comprobación de archivo se calcula como un valor de hash del recurso de ASP+ asociado (por ejemplo, el recurso de ASP+ 600 de la figura 6). La matriz comenzó en la línea 3 representa el comienzo de la información de estado para el siguiente nivel de jerarquía, en concreto el objeto de control de página 804 de la figura 8. El valor nulo en la línea 4 representa el primer elemento de la matriz de dos elementos de la línea 3, que es capaz de contener una información de estado en relación con las propiedades del propio objeto de página. En la realización ilustrada, no se registra información de estado alguna para el objeto de página, por lo tanto el valor es igual a “nulo”.

Tal como se muestra en la línea 5, el segundo elemento de la matriz de dos elementos de la línea 3 indica una tabla de hash que incluye una información de estado para los hijos del objeto de control de página (en concreto, los objetos de control 806, 808, 810 y 812 de la figura 8). La tabla de hash de la línea 5 incluye una entrada de tabla de hash, tal como se indica mediante el “1” en la etiqueta <H1>. Las etiquetas <K>, </K> en las líneas 6 y 8 engloban la clave para la entrada de tabla de hash, en concreto el valor entero “2” que se indica en la línea 7, que se corresponde con el índice del objeto de control de formulario 810 de la figura 8. El valor de _VIEWSTATE que se muestra en la figura 9 no incluye información de estado alguna para los objetos de control de literal 806, 808 y 812 de la figura 8. Como alternativa, si existieran otros objetos de control que tuvieran información de estado guardada u objetos de control hijos al mismo nivel de jerarquía que el objeto de control de formulario 810, entonces la tabla de hash a este nivel puede tener más de una entrada de tabla de hash para dar cabida a los objetos de control adicionales a este nivel, y la clave para cada entrada de tabla de hash adicional sería el índice del objeto de control correspondiente.

El valor de la entrada de tabla de hash es englobado por las etiquetas <V>, </V> de las líneas 9 y 24 e incluye otra matriz de dos elementos para definir el estado del objeto de control de formulario y sus hijos. El primer elemento de la matriz especifica la información de estado guardada para el objeto de control de formulario (es decir, que se representa mediante un valor nulo en la presente realización).

El segundo elemento especifica otra tabla de hash de una entrada que representa la información de estado para los hijos del objeto de control de formulario. Las etiquetas <K>, </K> en las líneas 13 y 15 engloban la clave para la única entrada de tabla de hash, en concreto el valor entero “1” que se indica en la línea 14, que se corresponde con el índice del objeto de control de etiqueta 816 de la figura 8. El valor de _VIEWSTATE que se muestra en la figura 9 no incluye información de estado alguna para los objetos de control de literal 814 y 818 de la figura 8.

El valor de la entrada de tabla de hash es englobado por las etiquetas <V>, </V> de las líneas 16 y 21 e incluye otra matriz de dos elementos para definir el estado del objeto de control de etiqueta y sus hijos. El primer elemento de la matriz especifica la información de estado guardada para una propiedad del objeto de control de etiqueta, que es el valor de cadena “InnerHtml = 16 / 5 / 99” que se especifica en la línea 18. El nombre de propiedad, “InnerHtml” es igual al valor de estado “16 / 5 / 99” y es de tipo “cadena”. El punto y coma representa el final del valor de estado para una propiedad dada. Si el objeto de control de etiqueta de la línea 18 incluyera propiedades adicionales para las cuales se guardó el estado, cada valor de datos de estado y su propiedad asociada estarían delimitados por un punto y coma de finalización. El valor nulo que se indica en la línea 19 al igual que en el segundo elemento de la matriz indica que el objeto de control de etiqueta no contiene hijo alguno. Las líneas 20 - 27 representan unas etiquetas de cierre para las etiquetas de inicio precedentes en la figura 9.

En una realización alternativa, se usa un formato de serialización que se denomina formato de Serialización Limitada de Objetos (LOS, *Limited Object Serialization*) en una estructura de estado transportable. En general, el formato de LOS especifica una tabla de hash de pares de nombre / valor para cada objeto de control, en la que cada entrada de hash contiene o bien una información de estado para una propiedad del objeto de control o bien una tabla de hash anidada de un objeto de control hijo.

La tabla 2 ilustra una gramática a modo de ejemplo del formato de LOS.

Tabla 2

Token	Gramática	Ejemplo
control object	• value type-table$_{opt}$ name-table$_{opt}$	• h<name1; value1>\t50System. Drawing. Color\n1BackColor
value	• typed-value • untyped-value • typed-array-value • untyped-array-value • untyped-hashtable-value	50<red> • <red> • a50<red; blue; green> • a<red; blue; green> • h<name1; value1>
typed-value	• type-ref value-list-start value-ref value-list-end	• 50<red>
typed-array-value	• array-modifier type-ref value-list-start array-value-ref value-list-end	• a50<red; blue; green>
untyped-value	• value-ref	• <red>
untyped-array-value	• array-modifier value-list-start array-value-ref value-list-end	• a<red; blue; green>
value-list-start	• <	
value-list-end	• >	
value-list-separator	• ;	
array-modifier	• **a**	
hashtable-modifier	• **h**	
untypedhashtablevalue	• hashtable-modifier value-list-start hashtable-value-ref value-list-end	• h<name1; value1>
value-ref	• string-value • bin-ref base 64-persisted-object	• \"Esto es un valor de cadena.\"
bin-ref	• escape-char **b**	• \b
array-value-ref	• value$_1$;[value$_2$;[value$_n$]]	• red; blue; green
hashtable-value-ref	• name-ref$_1$; value$_1$ [value-list-separator name-ref$_2$; value$_2$ [value-list-separator name-ref$_n$; value$_n$]]	• <name1; value1; name2; value2; name3; value3>
name-ref	• string-name-number • string-name	• 1 • BackColor
type-ref	• known-type-number • string-type-number • string-type	• 10 • 50 • System.Drawing.Color
type-table	• type-table-start string-type-number, string-type$_1$[value-list-separator string-type-number$_2$ string-type$_2$[value-list-separator string-type-number$_n$ string-typen;]]	• \t50System.Drawing.Color; 51System.Drawing.Font
name-table	• name-table-start string-number$_1$ string-name$_1$[value-list-separator string-name-number$_2$ string-name$_2$[value-list-separator string-name-number$_n$ string-name$_n$;]]	• \n1BackColor; 2ForeColor
known-type-number	• 0 1 2 3 4 5 6 7 8 9 • 10 11 12 13 14 15 16 17 18 19 • 20 21 22 23 24 25 26 272829 • 30 31 32 33 34 35 36 37 38 39 • 40 41 42 43 44 45 46 47 48 49	
type-table-start	• escape-char t	• \t
name-table-start	• escape-char n	• \n
string-terminator-char	• value-list-separator value-list-end ' ' " " \	

(continuación)

Token	Gramática	Ejemplo
escaped-string	• escape-char string-terminator-char	• \"
escape-char	• \	
string-number	• 0 1 2 3 4 5 6 7 8 9	
base 64-persisted-object	• Cualquier objeto que se haya sometido a Serialización Binaria y, a continuación, se haya codificado en base 64	
string-name	• Cualquier cadena que sea compensada por un delimitador de cadena	
string-type	• Cualquier cadena que coincida con un nombre de clase de COM+ y que sea compensada por un delimitador de cadena	
string-value	• Cualquier cadena que sea compensada por un delimitador de cadena que se pueda convertir usando la clase TypeConverter (convertidor de tipo) del tipo	

Como un primer ejemplo de cómo se puede usar el formato de LOS, considérese la entrada:

h<name1; value1; name2; value with \; escaped \> *"characters*\">

que define una tabla de hash de 2 pares de nombre / valor, en la que los valores son, todos ellos, cadenas. El primer valor se denomina “name1” y es igual a la cadena “value1”. El segundo valor se denomina “name1” y es igual a la cadena “*value with ; escaped > “characters*””.

Como un segundo ejemplo de cómo se puede usar el formato de LOS, considérese la entrada:

h<control1; h<1; 50<blue>; text; hello>control2; h<1; 50<red>; control3;>\t50System.Drawing.Color\n1BackColor

que define una tabla de hash que tiene tres entradas de hash: “control1”, “control2”, y “control3”. La primera entrada de hash se denomina “control1” e incluye una tabla de hash hija que tiene dos entradas, que son pares de nombre / valor. El nombre de la primera entrada en la tabla de hash de control1 usa el índice “1” para hacer referencia al nombre “BackColor”, que se define al final del ejemplo. El valor de la primera entrada es igual a “blue” (azul) y es de tipo “System.Drawing.Color”, lo que es especificado por el índice “50” y que se define cerca del final del ejemplo. El par de nombre / valor de la segunda entrada en la tabla de hash de control1 incluye el nombre “text” (texto) y el valor “hello” (hola). La entrada de hash de “control2” incluye una tabla de hash hija que tiene una entrada de hash, que también usa el índice “1” para hacer referencia al nombre “BackColor”. El valor es igual a “red” (rojo) y es de tipo “System.Drawing.Color”. La tabla de hash de “control3” está vacía. De una forma similar a la de la realización que se ha divulgado anteriormente, la jerarquía es especificada por el anidamiento de las tablas de hash; no obstante, se contemplan otros procedimientos de descripción de una jerarquía dentro del ámbito de la presente invención, incluyendo el uso de identificadores jerárquicos.

La figura 10 ilustra un diagrama de flujo de procedimiento para recibir una estructura de estado transportable y cargar la información de estado que está almacenada en la misma en unos objetos de control de una jerarquía de objetos de control en una realización de la presente invención. En la realización ilustrada, una estructura de estado transportable puede no enviarse desde el servidor (y, por lo tanto, no ser devuelta al servidor por el cliente) a menos que al menos una propiedad cambiara para al menos un objeto de control de lado de servidor. En ese sentido, el diagrama de flujo ilustrado supone que existe al menos un elemento de información de estado en la estructura de estado transportable recibida. En una realización alternativa, la estructura de estado transportable se puede enviar de ida y vuelta entre el servidor y el cliente, incluso si en la misma no existe información de estado alguna. En tal realización, una operación de decisión (que no se muestra) podría operar para abortar el procedimiento de gestión de estados si no existe información de estado alguna en la estructura de estado transportable.

La operación de recepción 1000 recibe una estructura de estado transportable a partir de un cliente, tal como en una solicitud de HTTP. La operación de marcado 1002 establece una indicación de estado inicial para todos los objetos de control en la jerarquía de lado de servidor. La operación de lectura 1004 lee la estructura de estado transportable recibida para extraer la información de estado y los correspondientes tipos de propiedad.

La operación de comprobación 1006 realiza una comprobación de integridad sobre la estructura de estado transportable (por ejemplo, usando el valor de _VIEWSTATEMAC). En una realización de la presente invención, la operación de comprobación 1006 implica (1) leer un código de integridad recibido a partir de la estructura de estado transportable; (2) calcular un código de integridad de sí misma a partir de al menos la información de estado que está incluida en la estructura de estado transportable; y (3) comparar el código de integridad recibido con el código de integridad recién calculado para determinar si la estructura de estado transportable se ha corrompido durante el

viaje de ida y vuelta al cliente. Si se determina que la estructura de estado transportable está corrompida, el código de lado de servidor puede responder para manejar la excepción, incluyendo al abortar la operación de carga, al continuar con el procesamiento en ausencia de información de estado o al generar un error.

Una operación de verificación 1008 verifica el código de comprobación de archivo que está incluido en la estructura de estado transportable. En una realización de la presente invención, la operación de verificación 1008 incluye (1) leer el código de comprobación de archivo recibido a partir del campo de _VIEWSTATE; (2) calcular su propio código de comprobación de archivo a partir de al menos los contenidos del almacén de recursos de ASP+ en el servidor; (3) y comparar el código de comprobación de archivo recibido con el recién calculado a partir del código de comprobación de archivo para determinar si la información de estado en la estructura de estado transportable se corresponde con la misma versión del recurso de ASP+ en el servidor. La operación de verificación 1008 se usa para verificar que el recurso de ASP+ en el servidor no cambió durante el viaje de ida y vuelta de la estructura de estado transportable al cliente. Si el recurso de ASP+ en el servidor sí cambió, entonces el servidor puede descartar la información de estado recibida, abortar la operación de carga y / o indicar un error al cliente. En una realización, la jerarquía de objetos de control se crea en su estado inicial y la operación de carga se aborta antes de que el servidor prosiga con el procesamiento de la solicitud. La operación de recorrido 1010 realiza un recorrido hasta el objeto de control de página de la jerarquía de objetos de control de lado de servidor.

La operación de análisis 1012 analiza información de estado para una jerarquía de objetos de control que se corresponde con un objeto de control en el campo de _VIEWSTATE anidado. En la primera iteración, el objeto de control correspondiente es el objeto de control de página hasta el que se realiza un recorrido en la operación de recorrido 1010. En iteraciones posteriores, el objeto de control correspondiente es un objeto de control descendente del objeto de control de página y se ubica mediante la operación de recorrido 1028. La operación de análisis 1012 puede extraer un valor de estado y convertir el valor, que se encuentra inicialmente en un formato de cadena cuando se analiza el mismo, en el tipo de propiedad dado. La operación de carga 1014 carga la información de estado que se analiza a partir del valor de _VIEWSTATE en una propiedad de un objeto de control. La operación de indicación 1016 establece una indicación de estado de cambio para el objeto de control actual. Esta indicación se puede usar más adelante para guardar solo el estado de aquellos objetos de control en los que el estado ha cambiado con respecto a un estado inicial, minimizando de ese modo el tamaño de la estructura de estado transportable. Como alternativa, se puede guardar toda la información de estado. Durante el manejo de datos de devolución, el manejo de eventos de devolución y las operaciones de enlace de datos en la jerarquía de objetos de control, los cambios que se hacen en el estado de un objeto de control también pueden dar como resultado el establecimiento de la indicación de estado cambiado para objetos de control individuales. En una realización de la presente invención, no obstante, cualquier indicación de cambio de datos que se establezca debido a las operaciones de enlace de datos se restablece de tal modo que el estado de esa propiedad no se registra en la estructura de estado transportable, minimizando adicionalmente de ese modo el tamaño de la estructura de estado transportable. El estado de una propiedad de este tipo depende de los datos que son especificados por la relación de enlace, que se actualizará en la operación de enlace de datos 512 de la figura 5.

La operación de decisión 1018 dirige el procesamiento a la operación 1022, si existe información de estado adicional para el objeto de control actual. Si esto es así, la operación 1022 analiza la información de estado para la siguiente propiedad de objeto de control en el objeto de control actual y dirige el procesamiento a la operación 1014. Si no se encuentra disponible información de estado adicional alguna para el objeto de control actual, la operación de decisión 1018 dirige el procesamiento a la operación de decisión 1020, que dirige el procesamiento a la operación de procesamiento 1024 si no se encuentra disponible otra información de estado de objeto de control en el valor de _VIEWSTATE. Si la operación de decisión 1020 determina que existen objetos de control adicionales para los cuales no se han procesado estados, el procesamiento avanza a la operación 1026, que analiza la información de estado para el siguiente objeto de control a partir del valor de _VIEWSTATE. La operación de recorrido 1028 realiza un recorrido hasta el siguiente objeto de control que se corresponde con la nueva información de estado que se analiza en la operación 1026. A continuación de lo anterior, el procesamiento avanza a la operación de carga 1014. Las operaciones de análisis recursivo de la figura 10 implementan una operación de deserialización en una realización de la presente invención.

La figura 11 ilustra un diagrama de flujo de procedimiento para guardar información de estado a partir de los objetos de control de una jerarquía de objetos de control en una estructura de estado transportable para la transmisión al cliente en una realización de la presente invención. La operación de inicialización 1100 inicializa una estructura de estado transportable. La operación de cálculo 1102 calcula un código de comprobación de archivo basándose en la versión actual del recurso de ASP+. En una realización de la presente invención, el código de comprobación de archivo está basado en un algoritmo de hash (por ejemplo, MD5 u otro) de los contenidos de recurso de ASP+, tales como meramente aplicar una función hash a los contenidos del archivo o algún componente especificado del archivo. La operación de carga 1104 carga el código de comprobación de archivo calculado en la estructura de estado transportable. La operación de recorrido 1106 realiza un recorrido hasta el objeto de página en la jerarquía de objetos de control.

En una realización de la presente invención, un objeto de control de lado de servidor se puede crear con una propiedad de “MaintainState” (mantener estado), que indica si los valores de propiedad del objeto de control (y sus hijos) se deberían guardar en la estructura de estado transportable. Si no es así, la operación de recorrido 1106

puede omitir el objeto de control de lado de servidor y sus hijos en este procedimiento. De lo contrario, si el estado del objeto de control se va a mantener de acuerdo con la propiedad de “MaintainState”, entonces la operación de recorrido 1106 realizará un recorrido hasta el objeto de control.

La operación de decisión 1108 dirige el procesamiento a la operación de lectura 1115, que lee el nuevo valor de estado y su tipo a partir de la propiedad del objeto de control. La operación 1110 registra el nuevo valor de estado y su tipo antes de avanzar a la operación 1111. Para los fines de este análisis, si no existe propiedad alguna dentro del objeto de control actual, la propiedad se representa mediante un valor “nulo” en el campo de _VIEWSTATE. Si la propiedad del objeto de control actual se encuentra aún en su estado inicial en la operación 1108, el procesamiento avanza a la operación de decisión 1111. Si existe otra propiedad dentro del objeto de control actual en la operación 1111, el procesamiento avanza a la operación de decisión 1108.

Si no existe otra propiedad en el objeto de control actual en la operación 1111, el procesamiento avanza a la operación de decisión 1112, que dirige el procesamiento a la operación de recorrido 1114 si existe otro objeto de control en la jerarquía. La operación de recorrido 1114 realiza un recorrido hasta el siguiente objeto de control en la jerarquía de objetos de control y avanza a la operación de decisión 1108 para acceder a la propiedad del objeto de control. Si no existe ningún otro objeto de control en la jerarquía en la operación 1112, el procesamiento avanza a la operación de serialización 1116 que serializa la nueva información de estado (incluyendo tipos de valor de estado) que se obtiene en las operaciones previas y la almacena en la estructura de estado transportable. La operación 1118 calcula un código de integridad (tal como mediante el uso de un algoritmo MD5), y la operación 1120 almacena el código de integridad en la estructura de estado transportable, tal como en el campo de _VIEWSTATEMAC. Se debería entender que las operaciones de código de integridad y las operaciones de código de comprobación de archivo son opcionales y se pueden omitir en una realización alternativa de la presente invención. La operación 1122 transmite la estructura de estado transportable al cliente, tal como en una respuesta de HTTP que contiene el código de HTML procesado que es generado por la jerarquía de objetos de control.

Las realizaciones de la invención que se describe en el presente documento son implementadas como etapas lógicas en uno o más sistemas informáticos. Las operaciones lógicas de la presente invención son implementadas (1) como una secuencia de etapas implementadas por procesador que se ejecutan en uno o más sistemas informáticos y (2) como unos módulos de máquina interconectados dentro de uno o más sistemas informáticos. La implementación es una cuestión de elección, que depende de los requisitos de funcionamiento del sistema informático que implementa la invención. Por consiguiente, se hace referencia a las operaciones lógicas que constituyen las realizaciones de la invención que se describe en el presente documento, de manera diversa, como operaciones, etapas, objetos o módulos.

La descripción, los ejemplos y los datos en lo que antecede proporcionan una descripción completa de la estructura y el uso de una realización de la invención. Debido a que pueden realizarse muchas realizaciones de la invención sin apartarse del ámbito de la invención, la invención reside en las reivindicaciones que se adjuntan en lo sucesivo en el presente documento.

REIVINDICACIONES

1. Un procedimiento de gestión de un estado de un objeto (314 - 322; 404 - 418) de control de lado de servidor que se corresponde con un elemento (106 - 110) de interfaz de usuario de lado de cliente que está incorporado en el código de lenguaje de creación que define una página (104) web que se representa en un cliente (100), comprendiendo el procedimiento:

crear el objeto de control de lado de servidor en la jerarquía de objetos (316) de control para procesar el elemento de interfaz de usuario de lado de cliente, en el que la jerarquía de objetos de control comprende una pluralidad de objetos de control de lado de servidor;
recibir (202) del cliente una estructura de estado transportable que incluye una información de estado que indica un valor de estado para al menos un objeto de control de lado de servidor en la jerarquía de objetos de control,
extraer la información de estado a partir de la estructura de estado transportable; y
cargar (1014) el valor de estado a partir de la información de estado en una propiedad del objeto de control de lado de servidor, si el valor de estado está asociado con el objeto de control de lado de servidor;
en el que la operación de extracción comprende:

extraer a partir de la estructura de estado transportable un valor de estado que está asociado con una propiedad del al menos un objeto de control de lado de servidor; e
identificar una ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en una información jerárquica dentro de la estructura de estado transportable;

y en el que la operación de carga comprende:

ubicar el objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en la ubicación jerárquica que se identifica en la operación de identificación;
recorrer la ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control; y
almacenar el valor de estado en la propiedad del objeto de control de lado de servidor.

2. El procedimiento de la reivindicación 1, en el que la estructura de estado transportable incluye adicionalmente un código de integridad recibido, y que comprende adicionalmente:

leer el código de integridad recibido a partir de la estructura de estado transportable;
calcular un código de integridad calculado a partir de la información de estado que está incluida en la estructura de estado transportable; y
comparar el código de integridad recibido con el código de integridad calculado para determinar si la estructura de estado transportable es válida.

3. El procedimiento de cualquiera de las reivindicaciones 1 a 2, que comprende adicionalmente:

inicializar (504) una propiedad del objeto de control de lado de servidor para tener un estado inicial;
establecer (1016) una indicación para representar que el estado de la propiedad del objeto de control de lado de servidor permanece sin cambios con respecto al estado inicial; y
cambiar la indicación para representar que la propiedad del objeto de control de lado de servidor ha cambiado con respecto al estado inicial, si en la propiedad se cargó el valor de estado a partir de la estructura de estado transportable.

4. El procedimiento de la reivindicación 3, que comprende adicionalmente:

establecer la indicación para indicar que la propiedad del objeto de control de lado de servidor permanece sin cambios con respecto al estado inicial, si la propiedad está enlazada por datos a un campo de un almacén de datos de lado de servidor.

5. El procedimiento de cualquiera de las reivindicaciones 1 a 4, que comprende adicionalmente:

recorrer cada objeto de control de lado de servidor en la jerarquía de objetos de control; y
almacenar la información de estado del objeto de control de lado de servidor en la estructura de estado transportable para la transmisión de vuelta al cliente, si la indicación representa que la información de estado del objeto de control de lado de servidor ha cambiado con respecto a su estado inicial.

6. El procedimiento de cualquiera de las reivindicaciones 1 a 5, que comprende adicionalmente:

transmitir (1122) la estructura de estado transportable al cliente en una respuesta con unos datos de lenguaje de creación que definen la página web.

7. El procedimiento de cualquiera de las reivindicaciones 1 a 6, que comprende adicionalmente:

calcular (1118) un código de integridad a partir de la información de estado que está incluida en la estructura de

estado transportable, sensible a la operación de almacenar la información de estado;
almacenar (1120) el código de integridad en la estructura de estado transportable; y
transmitir (1122) la estructura de estado transportable al cliente en una respuesta con unos datos de lenguaje de creación que definen la página web.

8. El procedimiento de cualquiera de las reivindicaciones 1 a 7, que comprende adicionalmente:

generar un código de lenguaje de creación que define al menos una porción de la página web a partir del al menos un objeto de control de lado de servidor en la jerarquía de objetos de control.

9. La página web definida por el código de lenguaje de creación y que es interpretable por un navegador que se está ejecutando en un sistema informático de cliente que está acoplado con un servidor web, siendo generado el código de lenguaje de creación en el servidor web por el procedimiento de la reivindicación 8, en la que el código de lenguaje de creación incluye la estructura de estado transportable que almacena información de estado para al menos un objeto de control de lado de servidor en la jerarquía de objetos de control.

10. Un medio de almacenamiento de programa informático legible por un sistema informático y que codifica un programa informático para ejecutar un procedimiento informático que gestiona un estado de un objeto (314 - 322; 404 - 418) de control de lado de servidor que se corresponde con un elemento (106 - 110) de interfaz de usuario de lado de cliente que está incorporado en el código de lenguaje de creación que define una página (104) web que se representa en un cliente (100), comprendiendo el procedimiento informático:

crear el objeto de control de lado de servidor en la jerarquía de objetos (316) de control para procesar el elemento de interfaz de usuario de lado de cliente, en el que la jerarquía de objetos de control comprende una pluralidad de objetos de control de lado de servidor,
recibir (202) del cliente una estructura de estado transportable que incluye información de estado que indica un valor de estado para al menos un objeto de control de lado de servidor en la jerarquía de objetos de control,
extraer la información de estado a partir de la estructura de estado transportable; y
cargar (1014) el valor de estado a partir de la información de estado en una propiedad del objeto de control de lado de servidor, si el valor de estado está asociado con el objeto de control de lado de servidor,
en el que la operación de extracción comprende:

extraer a partir de la estructura de estado transportable un valor de estado que está asociado con una propiedad del al menos un objeto de control de lado de servidor; e
identificar una ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en una información jerárquica dentro de la estructura de estado transportable;

y en el que la operación de carga comprende:

ubicar el objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en la ubicación jerárquica identificada en la operación de identificación;
recorrer la ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control; y
almacenar el valor de estado en la propiedad del objeto de control de lado de servidor.

11. Una señal de datos informáticos que se materializa en una onda portadora por medio de un sistema informático y que codifica un programa informático para ejecutar un procedimiento informático que gestiona un estado de un objeto (314 - 322; 404 - 418) de control de lado de servidor que se corresponde con un elemento (106 - 110) de interfaz de usuario de lado de cliente que está incorporado en el código de lenguaje de creación que define una página (104) web que se representa en un cliente (100), comprendiendo el procedimiento informático:

crear el objeto de control de lado de servidor en la jerarquía de objetos (316) de control para procesar el elemento de interfaz de usuario de lado de cliente, en la que la jerarquía de objetos de control comprende una pluralidad de objetos de control de lado de servidor;
recibir (202) del cliente una estructura de estado transportable que incluye una información de estado que indica un valor de estado para al menos un objeto de control de lado de servidor en la jerarquía de objetos de control,
extraer la información de estado a partir de la estructura de estado transportable; y
cargar (1014) el valor de estado a partir de la información de estado en una propiedad del objeto de control de lado de servidor, si el valor de estado está asociado con el objeto de control de lado de servidor,
en la que la operación de extracción comprende:

extraer a partir de la estructura de estado transportable un valor de estado que está asociado con una propiedad del al menos un objeto de control de lado de servidor; e
identificar una ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en una información jerárquica dentro de la estructura de estado transportable,

y en la que la operación de carga comprende:

ubicar el objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en la ubicación jerárquica que se identifica en la operación de identificación;
recorrer la ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control; y
almacenar el valor de estado en la propiedad del objeto de control de lado de servidor.

12. Un producto de programa informático que codifica un programa informático para ejecutar, en un sistema informático, un procedimiento informático para gestionar un estado de una pluralidad de objetos (314 - 322; 404 - 418) de control de lado de servidor que se crea en una jerarquía de objetos (316) de control y que se corresponde con una pluralidad de elementos (106 - 110) de interfaz de usuario de lado de cliente que están incorporados en el código de lenguaje de creación que define una página (104) web que se representa en un cliente (100), comprendiendo el procedimiento informático:

recibir (202) del cliente una estructura de estado transportable que incluye información de estado que está asociada con uno o más objetos de control de lado de servidor en la jerarquía de objetos de control;
deserializar la información de estado para extraer un valor de estado, un tipo de datos de propiedad asociado y una información jerárquica para una propiedad de un objeto de control de lado de servidor;
identificar una ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en la información jerárquica;
ubicar el objeto de control de lado de servidor dentro de la jerarquía de objetos de control basándose en la ubicación jerárquica que se identifica en la operación de identificación;
recorrer la ubicación jerárquica del objeto de control de lado de servidor dentro de la jerarquía de objetos de control; y
cargar (1014) el valor de estado en la propiedad del objeto de control de lado de servidor.

13. El producto de programa informático de la reivindicación 12, en el que el procedimiento informático comprende adicionalmente:

convertir el valor de estado para tener el tipo de datos de propiedad asociado antes de la operación de carga.

14. El producto de programa informático de la reivindicación 12 o 13, en el que el procedimiento informático comprende adicionalmente:

inicializar (504) la propiedad del objeto de control de lado de servidor para tener un estado inicial;
establecer (1016) una indicación para representar que la propiedad del objeto de control de lado de servidor permanece sin cambios con respecto al estado inicial; y
cambiar la indicación para representar que la propiedad del objeto de control de lado de servidor ha cambiado con respecto al estado inicial, si en la propiedad se cargó el valor de estado a partir de la estructura de estado transportable.

15. El producto de programa informático de cualquiera de las reivindicaciones 12 a 14, en el que el procedimiento informático comprende adicionalmente recorrer uno o más de los objetos de control de lado de servidor en la jerarquía de objetos de control y, para cada uno de los objetos de control de lado de servidor:

extraer un valor de propiedad a partir del objeto de control de lado de servidor que está asociado con la indicación que representa que el estado del objeto de control de lado de servidor ha cambiado con respecto a su estado inicial; y
serializar el valor de propiedad en la estructura de estado transportable con los valores de propiedad a partir de otros objetos de control de lado de servidor en la jerarquía de objetos de control para la transmisión de vuelta al cliente.

16. El producto de programa informático de cualquiera de las reivindicaciones 12 a 15, en el que el procedimiento informático comprende adicionalmente:

extraer un tipo de datos de propiedad a partir del objeto de control de lado de servidor que está asociado con el valor de propiedad; y
serializar el tipo de datos de propiedad en la estructura de estado transportable con el valor de propiedad del objeto de control de lado de servidor para la transmisión de vuelta al cliente.

17. El producto de programa informático de cualquiera de las reivindicaciones 12 a 16, en el que el procedimiento informático comprende adicionalmente:

almacenar, en la estructura de estado transportable, una información jerárquica en relación con el objeto de control de lado de servidor en la jerarquía de objetos de control para la transmisión al cliente.

18. El producto de programa informático de cualquiera de las reivindicaciones 12 a 17, en el que el procedimiento informático comprende adicionalmente:

transmitir (1122) la estructura de estado transportable al cliente en una respuesta con unos datos de lenguaje de creación que definen una página web.

19. El producto de programa informático de cualquiera de las reivindicaciones 12 a 18, en el que el procedimiento informático comprende adicionalmente:

calcular (1118) un código de integridad a partir de la información de estado que está almacenada en la estructura de estado transportable, sensible a la operación de serialización;
almacenar (1120) el código de integridad en la estructura de estado transportable; y
transmitir (1122) la estructura de estado transportable al cliente en una respuesta con unos datos de lenguaje de creación que definen una página web.

20. El producto de programa informático de cualquiera de las reivindicaciones 12 a 19, en el que la estructura de estado transportable incluye adicionalmente un código de integridad recibido, y que comprende adicionalmente:

leer el código de integridad recibido a partir de la estructura de estado transportable;
calcular un código de integridad calculado a partir de la información de estado que está incluida en la estructura de estado transportable; y
comparar el código de integridad recibido con el código de integridad calculado para determinar si la estructura de estado transportable es válida.

21. Un sistema manipulador que se ejecuta en un ordenador de servidor para procesar una solicitud que incluye una estructura de estado transportable que almacena un valor de estado y un tipo de datos a partir de un ordenador (100) de cliente, comprendiendo el sistema manipulador:

una jerarquía (316) de objetos (314 - 322; 404 - 418) de control de lado de servidor que son creados por el manipulador y que están asociados con los elementos (106 - 110) de interfaz de usuario de lado de cliente, incluyendo al menos uno de los objetos de control de lado de servidor una propiedad, en el que la jerarquía de objetos de control comprende una pluralidad de objetos de control de lado de servidor;
un módulo de lectura que está adaptado para extraer el valor de estado, una información jerárquica asociada y el tipo de datos a partir de una estructura de estado transportable;
un módulo de recorrido que está adaptado para ubicar el objeto de control dentro de la jerarquía de objetos de control, basándose en la información jerárquica extraída;
un módulo de carga que está adaptado para almacenar el valor de estado que se recibe en la estructura de estado transportable en la propiedad del objeto de control de lado de servidor ubicado, basándose en el tipo de datos; y
un módulo de guardado que está adaptado para almacenar la propiedad y el tipo de datos en una estructura de estado transportable.

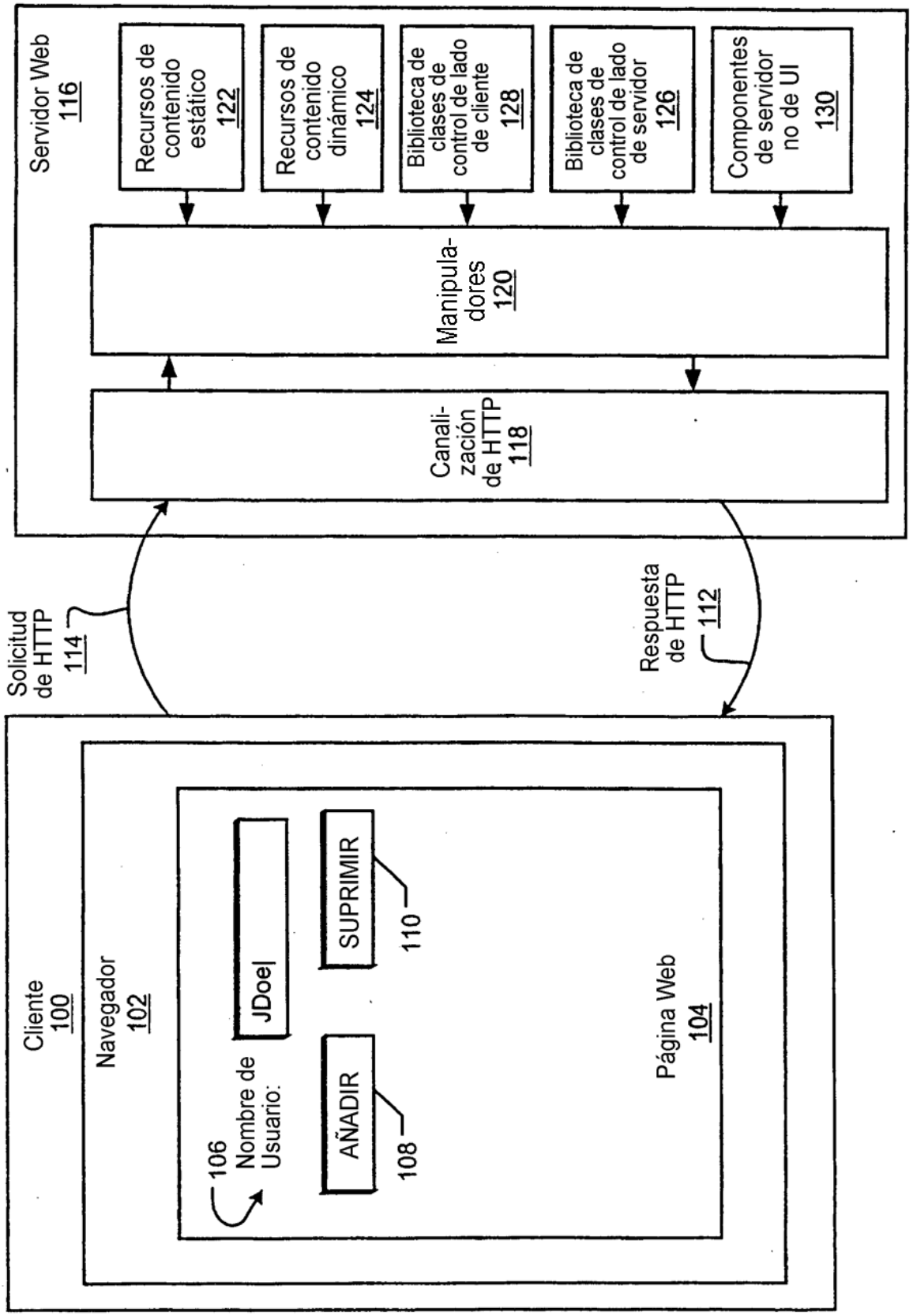

Cliente
100
Navegador
102
106
Nombre de Usuario:
JDoe|
AÑADIR
108
SUPRIMIR
110
Página Web
104
Solicitud de HTTP
114
Respuesta de HTTP
112
Servidor Web
116
Canali-zación de HTTP
118
Manipula-dores
120
Recursos de contenido estático
122
Recursos de contenido dinámico
124
Biblioteca de clases de control de lado de cliente
128
Biblioteca de clases de control de lado de servidor
126
Componentes de servidor no de UI
130

FIG. 1

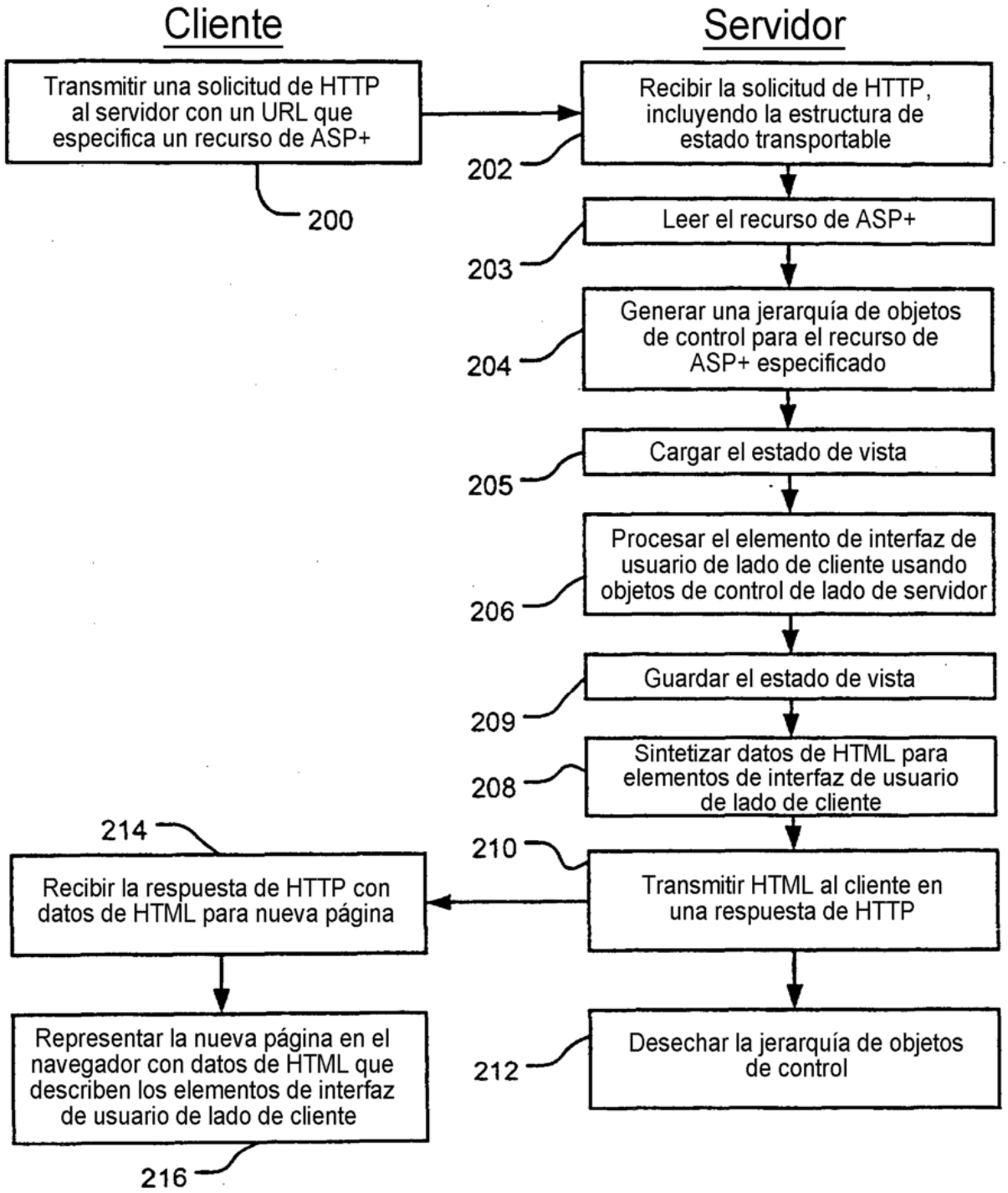

Cliente
Servidor
Transmitir una solicitud de HTTP al servidor con un URL que especifica un recurso de ASP+
200
Recibir la solicitud de HTTP, incluyendo la estructura de estado transportable
202
Leer el recurso de ASP+
203
Generar una jerarquía de objetos de control para el recurso de ASP+ especificado
204
Cargar el estado de vista
205
Procesar el elemento de interfaz de usuario de lado de cliente usando objetos de control de lado de servidor
206
Guardar el estado de vista
209
Sintetizar datos de HTML para elementos de interfaz de usuario de lado de cliente
208
210
Transmitir HTML al cliente en una respuesta de HTTP
212
Desechar la jerarquía de objetos de control
214
Recibir la respuesta de HTTP con datos de HTML para nueva página
Representar la nueva página en el navegador con datos de HTML que describen los elementos de interfaz de usuario de lado de cliente
216

FIG. 2

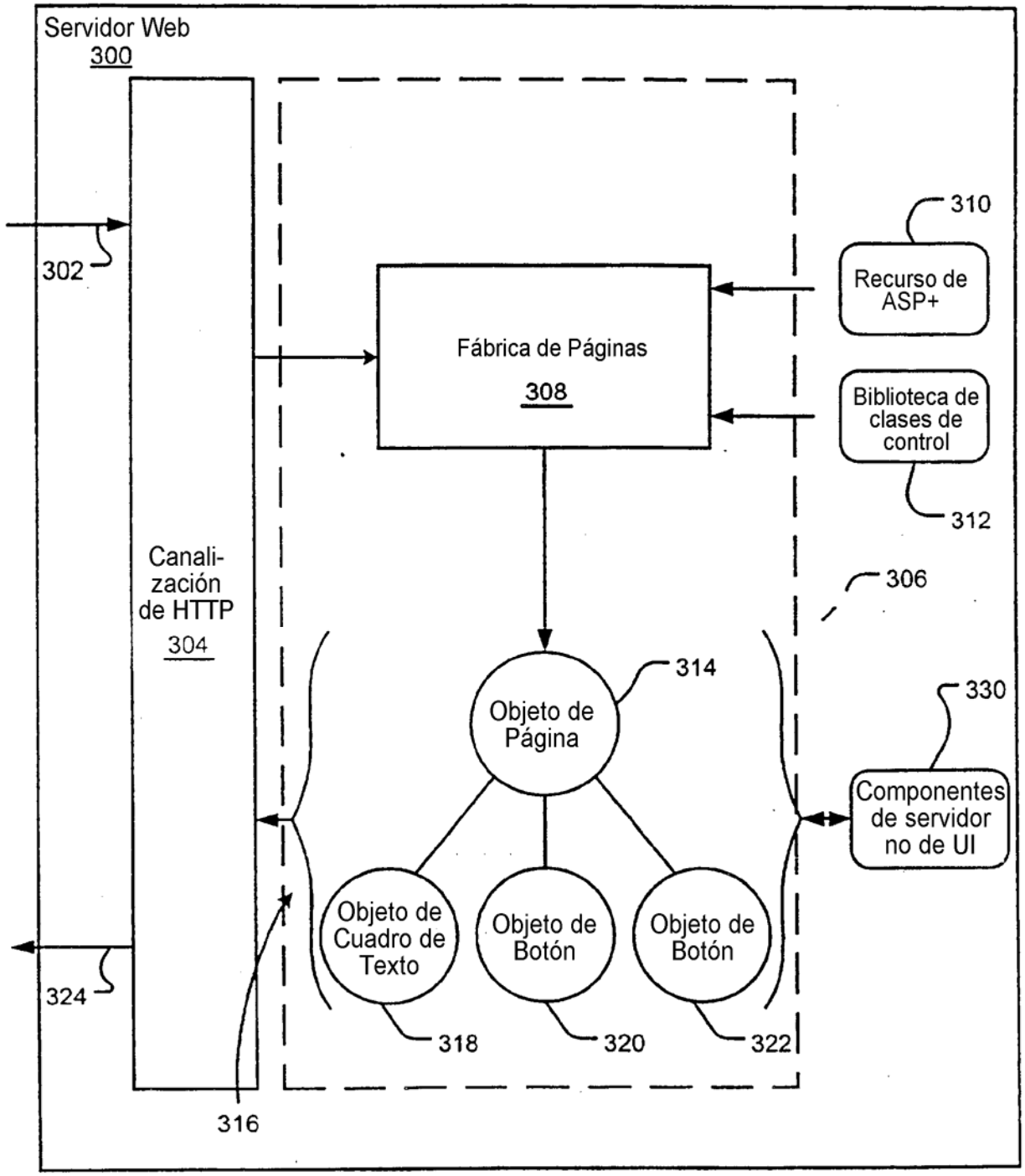

Servidor Web
300
302
Canali-
zación
de HTTP
304
Fábrica de Páginas
308
310
Recurso de
ASP+
Biblioteca de
clases de
control
312
306
314
Objeto de
Página
330
Componentes
de servidor
no de UI
Objeto de
Cuadro de
Texto
Objeto de
Botón
Objeto de
Botón
318
320
322
324
316

FIG. 3

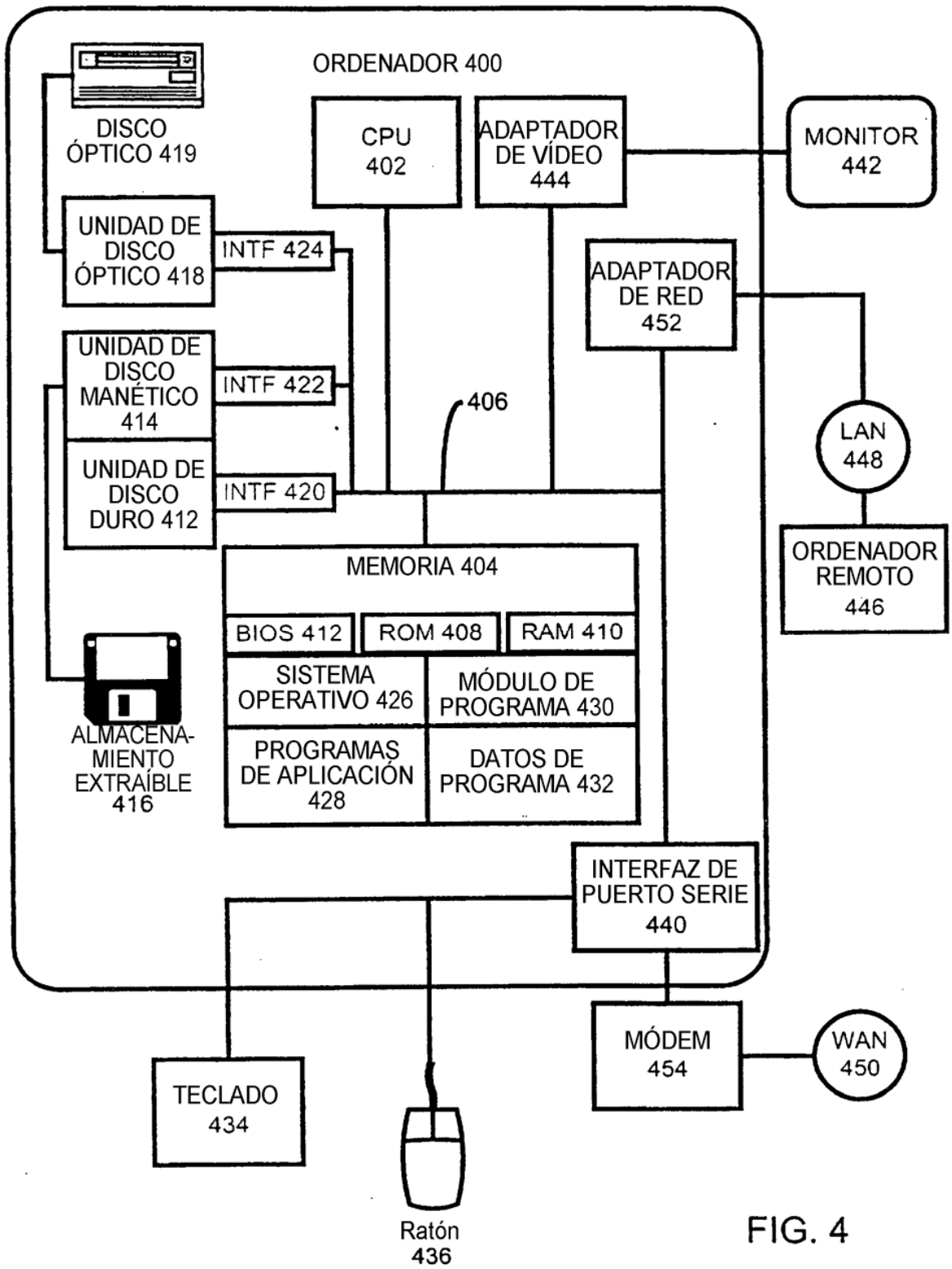

ORDENADOR 400
DISCO ÓPTICO 419
CPU 402
ADAPTADOR DE VÍDEO 444
MONITOR 442
UNIDAD DE DISCO ÓPTICO 418
INTF 424
ADAPTADOR DE RED 452
UNIDAD DE DISCO MANÉTICO 414
INTF 422
406
LAN 448
UNIDAD DE DISCO DURO 412
INTF 420
ORDENADOR REMOTO 446
MEMORIA 404
BIOS 412
ROM 408
RAM 410
SISTEMA OPERATIVO 426
MÓDULO DE PROGRAMA 430
ALMACENA-MIENTO EXTRAÍBLE 416
PROGRAMAS DE APLICACIÓN 428
DATOS DE PROGRAMA 432
INTERFAZ DE PUERTO SERIE 440
MÓDEM 454
WAN 450
TECLADO 434
Ratón 436

FIG. 4

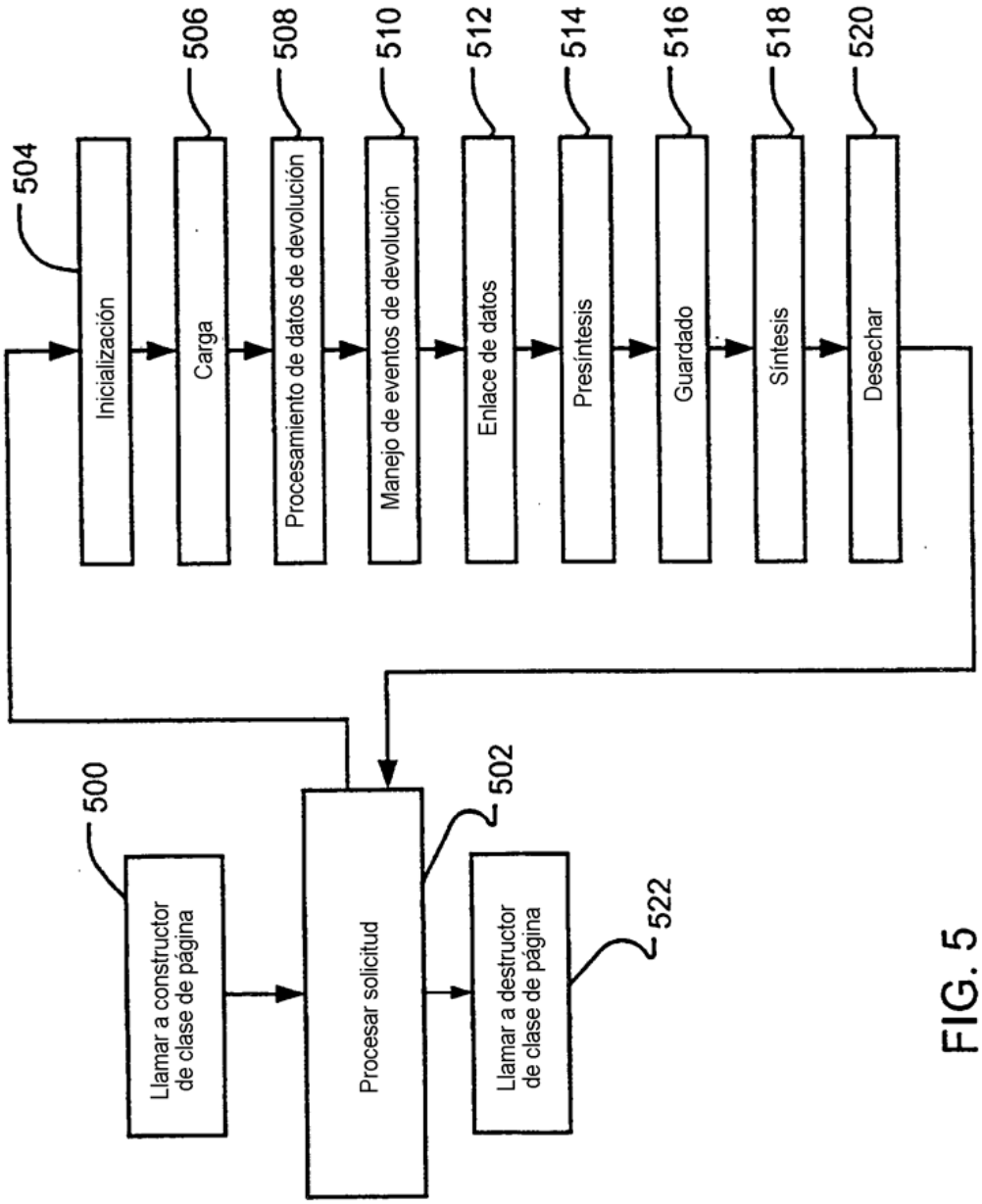
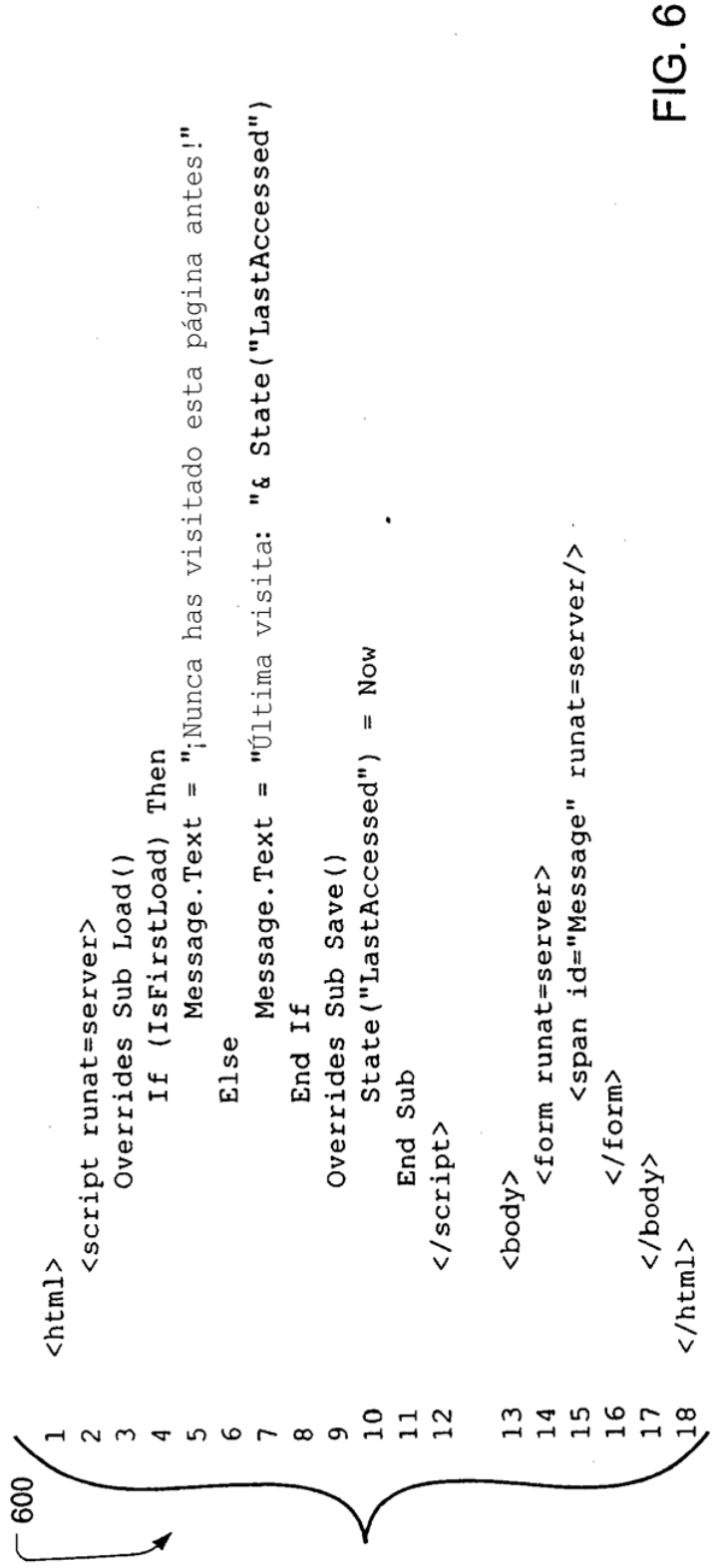

504
Inicialización
506
Carga
508
Procesamiento de datos de devolución
510
Manejo de eventos de devolución
512
Enlace de datos
514
Presíntesis
516
Guardado
518
Síntesis
520
Desechar
500
Llamar a constructor de clase de página
502
Procesar solicitud
522
Llamar a destructor de clase de página

FIG. 5


600
<html>
<script runat=server>
Overrides Sub Load()
If (IsFirstLoad) Then
Message.Text = "¡Nunca has visitado esta página antes!"
Else
Message.Text = "Última visita: "& State("LastAccessed")
End If
Overrides Sub Save()
State("LastAccessed") = Now
End Sub
</script>
<body>
<form runat=server>

</form>
</body>
</html>

FIG. 6

700
<form>
<input type=hidden name="__VIEWSTATE"value="<a2><s>-5120857414</s><a2>
<n><H1><K>2</K><V><a2><n><H1><K>1</K><V><a2><s>InnerHtml=5/16/99;
</s><n></V></H></V></H>">
<input type=hidden name ="__VIEWSTATEMAC" value="434343333433">
Last Post Back: 5/6/99
</form>

FIG. 7

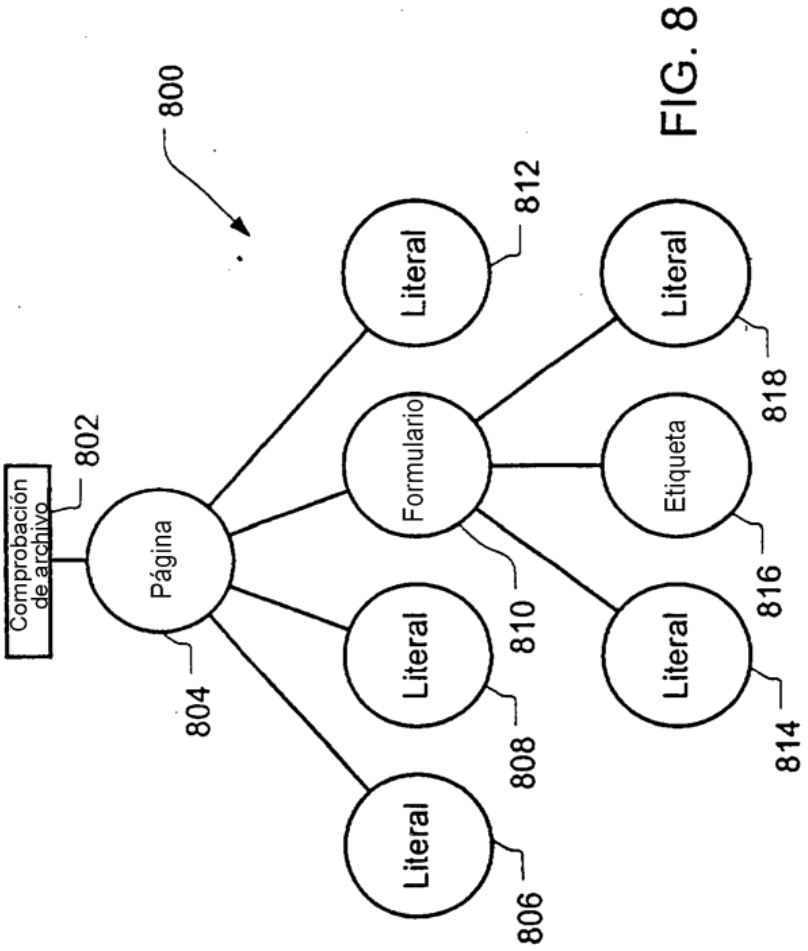

800
Comprobación de archivo
802
Página
804
Literal
806
Literal
808
Literal
810
Formulario
812
Literal
814
Etiqueta
816
Literal
818

FIG. 8

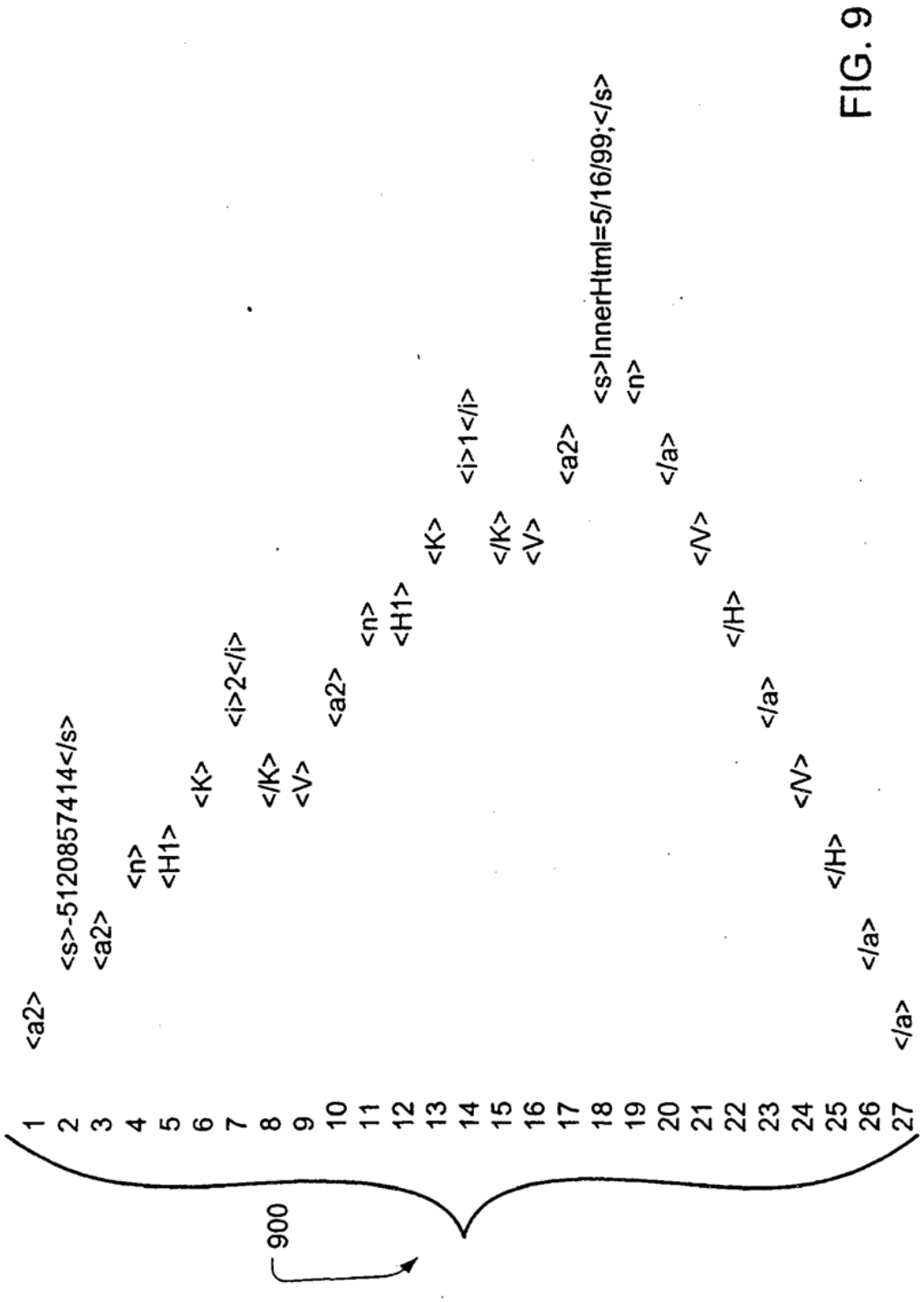

1 <a2>
2 <s>-5120857414</s>
3 <a2>
4 <n>
5 <H1>
6 <K>
7 2
8 </K>
9 <V>
10 <a2>
11 <n>
12 <H1>
13 <K>
14 1
15 </K>
16 <V>
17 <a2>
18 <s>InnerHtml=5/16/99;</s>
19 <n>
20
21 </V>
22 </H>
23
24 </V>
25 </H>
26
27
900

FIG. 9

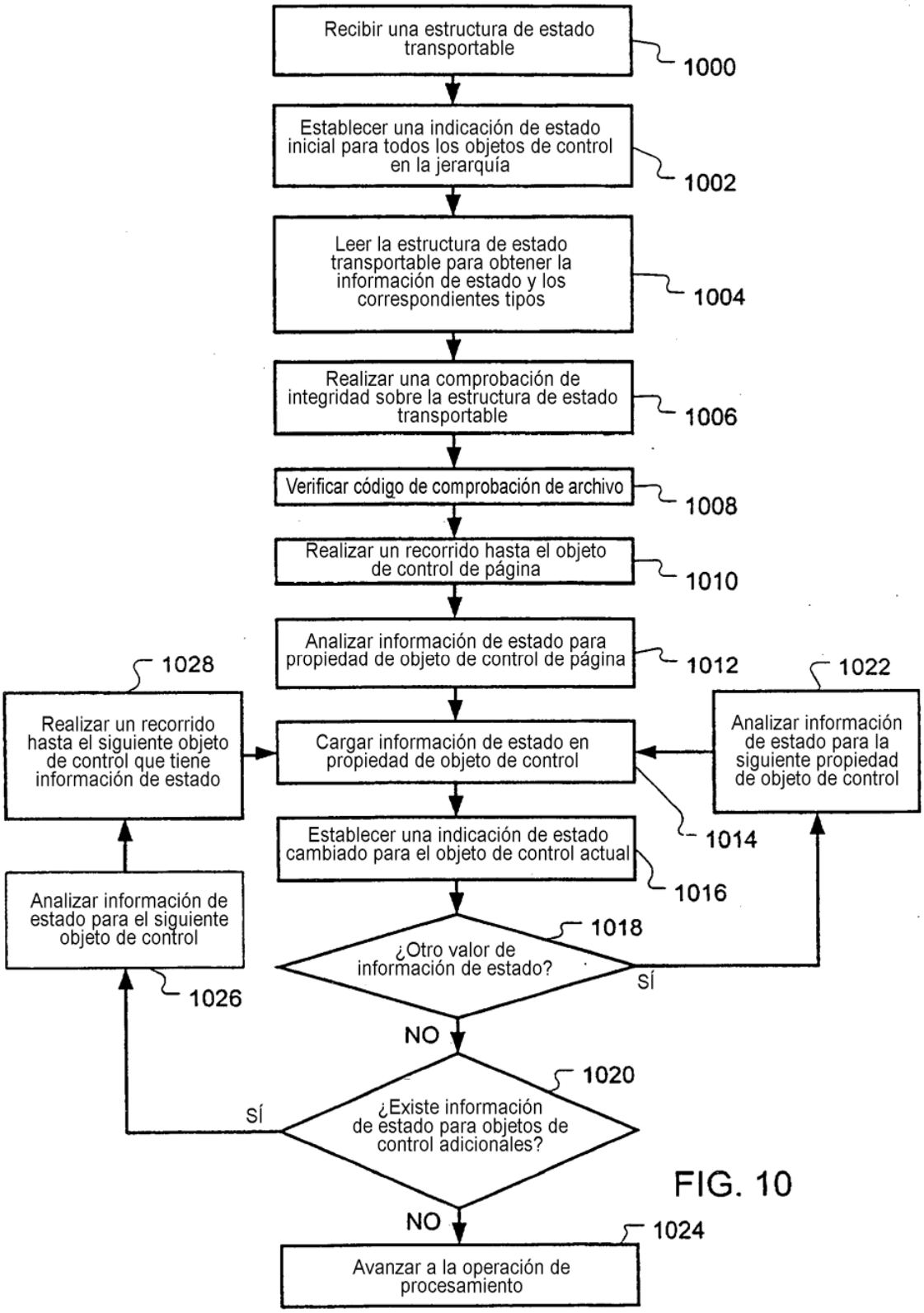

Recibir una estructura de estado transportable
1000
Establecer una indicación de estado inicial para todos los objetos de control en la jerarquía
1002
Leer la estructura de estado transportable para obtener la información de estado y los correspondientes tipos
1004
Realizar una comprobación de integridad sobre la estructura de estado transportable
1006
Verificar código de comprobación de archivo
1008
Realizar un recorrido hasta el objeto de control de página
1010
Analizar información de estado para propiedad de objeto de control de página
1012
1028
Realizar un recorrido hasta el siguiente objeto de control que tiene información de estado
Cargar información de estado en propiedad de objeto de control
1022
Analizar información de estado para la siguiente propiedad de objeto de control
1014
Establecer una indicación de estado cambiado para el objeto de control actual
1016
Analizar información de estado para el siguiente objeto de control
1026
1018
¿Otro valor de información de estado?
SÍ
NO
1020
¿Existe información de estado para objetos de control adicionales?
SÍ
NO
1024
Avanzar a la operación de procesamiento

FIG. 10

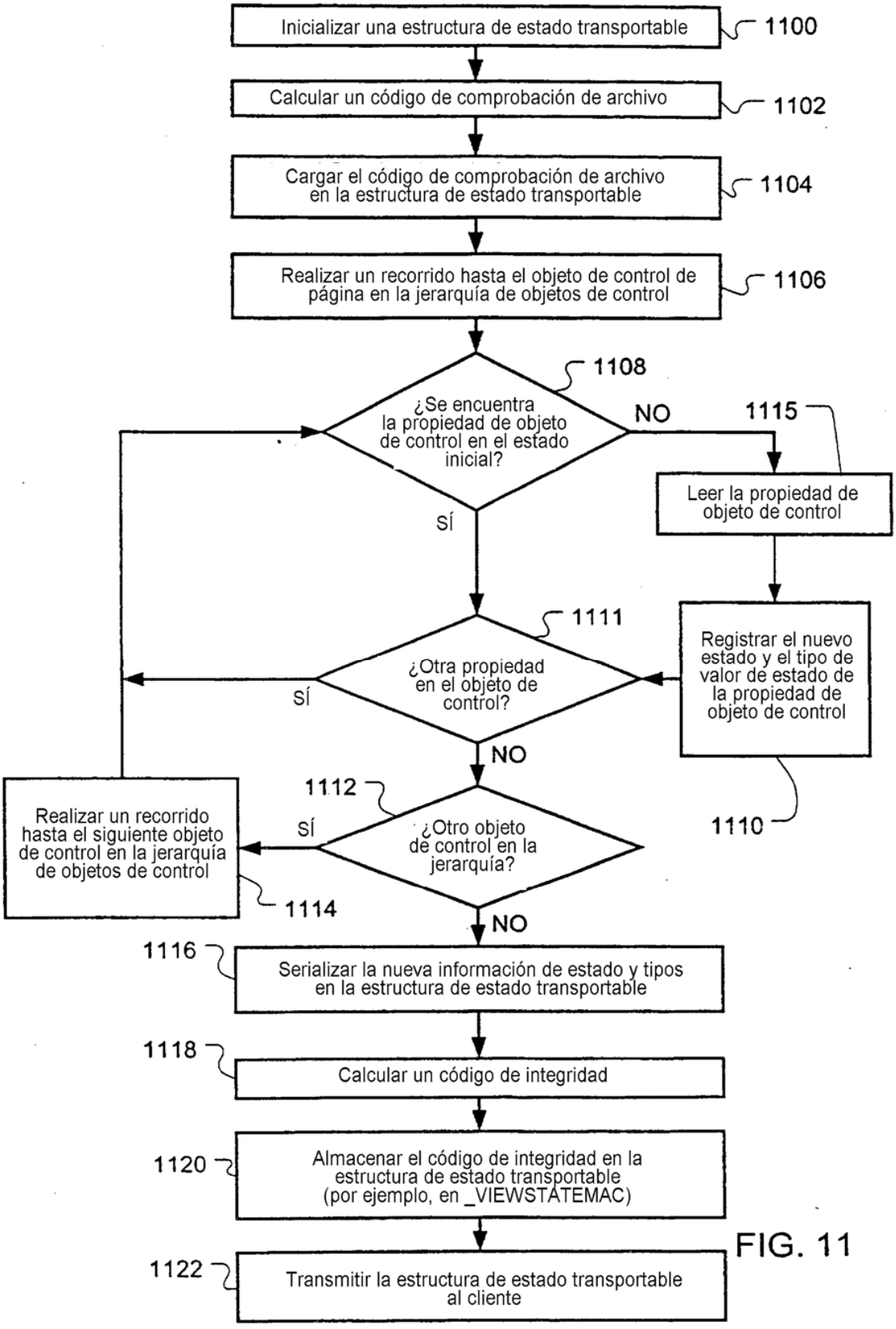

Inicializar una estructura de estado transportable
1100
Calcular un código de comprobación de archivo
1102
Cargar el código de comprobación de archivo en la estructura de estado transportable
1104
Realizar un recorrido hasta el objeto de control de página en la jerarquía de objetos de control
1106
1108
¿Se encuentra la propiedad de objeto de control en el estado inicial?
NO
1115
Leer la propiedad de objeto de control
SÍ
1111
¿Otra propiedad en el objeto de control?
SÍ
Registrar el nuevo estado y el tipo de valor de estado de la propiedad de objeto de control
NO
1110
1112
¿Otro objeto de control en la jerarquía?
SÍ
Realizar un recorrido hasta el siguiente objeto de control en la jerarquía de objetos de control
1114
NO
1116
Serializar la nueva información de estado y tipos en la estructura de estado transportable
1118
Calcular un código de integridad
1120
Almacenar el código de integridad en la estructura de estado transportable (por ejemplo, en _VIEWSTATEMAC)
1122
Transmitir la estructura de estado transportable al cliente

FIG. 11