

MODELOS ASÍNCRONOS

REFERENCIA CRUZADA A SOLICITUD RELACIONADA

[0001] La presente solicitud reivindica prioridad a la solicitud de patente no provisional de Estados Unidos con número de serie 14/843.792 presentada el 2 de septiembre de 2015 y la solicitud de patente provisional de Estados Unidos con número de serie 62/046.109, presentada el 4 de septiembre de 2014, cada una se incorpora aquí en su totalidad como referencia.

ANTECEDENTES

[0002] En la programación orientada a objetos, el patrón Modelo-Vista-Controlador (MVC) es un patrón de arquitectura de software que divide un programa de aplicación en partes de componentes de modelo, vista y controlador. Esto crea una separación entre la representación interna de los datos y la manera en que los datos se presentan al usuario (o entrada aceptada por parte del usuario).

[0003] El componente de modelo es la representación interna de los datos, es decir, un objeto que representa algunos datos. El componente de vista es un objeto que proporciona una representación de esos datos para el usuario, es decir, la vista puede ser considerada una visualización del estado del modelo. El objeto de componente de controlador acepta la entrada del usuario y, por ejemplo, permite cambiar el estado del modelo.

[0004] Mientras que el modelo tradicional MVC funciona bien para muchos tipos de programas de aplicación, hay problemas con él cuando los datos que un modelo y/o vista necesita no están fácilmente disponibles. Por ejemplo, a veces los datos necesitan ser recuperados de una fuente de red, mediante el cual los modelos y puntos de vista tienen que esperar a los datos.

RESUMEN

[0005] Se provee este resumen para introducir una selección de conceptos de representación en una forma simplificada que se describen más adelante en la descripción detallada. Este resumen no pretende identificar las características clave o características esenciales de la materia reivindicada, ni está destinado a ser utilizado en cualquier forma que pueda limitar el alcance de la materia reivindicada.

[0006] En resumen, la tecnología descrita en el presente documento está dirigido a recibir, en un modelo de vista, una solicitud de datos específicos de vista de una vista, y en el modelo de vista, haciendo una o más peticiones asíncronas a un modelo de datos para la información correspondiente a la solicitud de datos específicos de vista. Una promesa se devuelve desde la vista del modelo a la vista en respuesta a la solicitud de datos de vista especificada. El modelo de datos obtiene un conjunto de datos que incluye la información y proporciona la información al modelo de vista en respuesta a una o más peticiones asincrónicas. El modelo de vista devuelve los datos de la vista especificada, en base a la información, a la vista de cumplir la promesa.

[0007] Otras ventajas pueden resultar evidentes a partir de la siguiente descripción detallada cuando se toma en conjunción con los dibujos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0008] La presente tecnología se ilustra a modo de ejemplo y sin limitación en las figuras adjuntas en las que los números de referencia similares indican elementos similares y en los que:

[0009] La figura 1 es un diagrama de bloques que muestra un ejemplo de configuración de los componentes (incluyendo vista y los objetos del modelo asíncronos) que pueden ser utilizados para devolver datos de vista específica a una vista solicitante, de acuerdo con una o más implementaciones del ejemplo.

[0010] La figura 2A es un diagrama de bloque que muestra un ejemplo de configuración de los componentes donde dos modelos de vista solicitan y obtienen información del mismo modelo de datos, de acuerdo con una o más implementaciones del ejemplo.

5 [0011] La figura 2B es un diagrama de bloque que muestra un ejemplo de configuración de los componentes donde una vista (o múltiples vistas) se comunican con diferentes modelos de vista donde cada modelo de vista, tiene la misma interfaz implementada por diferente implementación del proveedor, según una o más implementaciones del ejemplo.

10 [0012] La figura 3 es un diagrama de bloques que muestra una secuencia de ejemplo de las solicitudes y las respuestas de datos, incluyendo promesas para peticiones asíncronas, de acuerdo con una o más implementaciones del ejemplo.

[0013] La figura 4 es una representación de ejemplo de modelos incluyendo interfaces y proveedores que están configurados para devolver datos de vista
15 especificados solicitados a una vista, de acuerdo con una o más implementaciones del ejemplo.

[0014] La figura 5 es una representación de un diagrama de ejemplo de secuencia/flujo de datos donde varios componentes del modelo operan para solicitar y recibir datos para devolver datos de vista especificados a una vista, de acuerdo
20 con una o más implementaciones del ejemplo.

[0015] La figura 6 es un diagrama de bloque de ejemplo que representa componentes de un sistema UI de plataforma independiente en la que pueden incorporarse una vista, modelo de vista y modelo de datos, de acuerdo con una o más implementaciones de ejemplo.

25 [0016] La figura 7 es un diagrama de flujo que muestra un ejemplo de etapas que pueden ser tomadas por un objeto de vista para hacer peticiones asíncronas para datos de vista-especificados de un modelo de vista y recibir los datos de vista

especificados en respuesta, de acuerdo con una o más implementaciones de ejemplo.

[0017] Las figuras 8 y 9 comprenden un diagrama de flujo que muestra un ejemplo de etapas que pueden ser tomadas por un modelo de vista para comunicarse con una vista y un modelo de datos para devolver datos de vista especificados a una vista solicitante, de acuerdo con una o más implementaciones de ejemplo.

[0018] La figura 10 es un diagrama de flujo que muestra un ejemplo de etapas que pueden ser tomadas por un modelo de datos para obtener datos de una fuente de datos, que comprende un servicio de caché y/o de red, para responder a una solicitud de un modelo de vista, de acuerdo con una o más implementaciones de ejemplo.

[0019] La figura 11 es un diagrama de bloques que representa un entorno de computación de ejemplo en el que se pueden incorporar aspectos de la materia descrita en este documento.

DESCRIPCIÓN DETALLADA

[0020] La tecnología descrita en el presente documento se dirige a tener modelos que se rellena con datos de forma asíncrona. A través de la tecnología descrita en el presente documento, un modelo puede ser creado inicialmente con todos, ninguno, o algunos de sus datos, mientras que los elementos de usuario en interfaz (UL) que comprenden componentes de vista pueden ser creados que dependen de los datos que un programa de aplicación no tienen todavía. Si un modelo carece de los datos, el modelo contiene la lógica para recuperar los datos que faltan, por ejemplo, desde otro modelo intermedio o desde una fuente de datos remota tal como un servidor de alojamiento de los datos para un cliente. Para este fin, cuando los datos todavía no están disponibles, una respuesta que comprende una "promesa" o similar (a veces referido como un "futuro" o "retardo" en entornos de programación asíncronos) puede ser devuelto, por ejemplo, creado como un objeto. Tenga en

cuenta que, como se usa en el presente documento, "promesa" es un término genérico que representa dicha respuesta, y por ejemplo puede estar relacionado con, pero no se limita a, JavaScript® Promise. Como también se usa en este documento, una promesa es "cumplida" cuando se devuelven los datos correspondientes (a veces referidos como "resolver" o "unión" de la promesa en entornos de programación asíncronos). Una promesa puede ser rechazada en el caso de un error.

[0021] La separación del modelo y la vista proporcionada por la tecnología permite otras partes del código, tales como las vistas en la interfaz de usuario, para interactuar con los modelos de forma asíncrona sin conocer los detalles de cómo se fue a buscar los datos, o cómo se almacenan localmente, y permite que las vistas se actualicen cuando los datos están disponibles en el futuro.

[0022] Los modelos, en este sistema, pueden comprender interfaces de modelo que definen los datos que el modelo es capaz de resolver. El modelo define la forma de los datos utilizados por la aplicación, y define lo que los datos son, pero no de donde vienen los datos. Los proveedores de datos son implementaciones asociadas con las interfaces de modelo, que contienen el código necesario para rellenar la interfaz mediante la realización de la operación u operaciones necesarias, de forma asíncrona.

[0023] Los proveedores de datos pueden tener que realizar alguna traducción entre la forma de los datos de la fuente de datos, y la forma definida por la interfaz modelo. Cuando se solicita datos de un modelo, el proveedor subyacente puede devolver datos en caché si está disponible o puede hacer las solicitudes de servicio necesarias para recuperar los datos. Los proveedores de datos saben dónde se necesitan los datos, y cómo obtener esos datos.

[0024] Como parte de la ejecución de la solicitud de servicio asíncrona para conseguir un pedazo de datos solicitados, un proveedor puede recibir una respuesta que contiene los datos adicionales que pueden ser utilizados por su interfaz. El

proveedor hace estas solicitudes sólo cuando sea necesario, y puede almacenar en caché los datos adicionales cuando se resuelve una solicitud para reducir el número de veces que el cliente tiene que obtener los datos de una fuente de datos a distancia (en red).

5 [0025] La tecnología descrita en el presente documento hace uso de este modelo asincrónico en capas separadas de la aplicación. En la capa de servicio (datos), las interfaces de modelo de datos y proveedores de datos definen como se deben ver los datos en los cuales la aplicación del cliente funcionará, y a dónde ir para obtener esos datos, (por ejemplo, que servicios a distancia tener, y cómo traducir sus
10 respuestas). En la capa de vista UI (diseño), las vistas definen los elementos UI y tienen sus propios modelos de vista. Estos "modelos de vista" definen los datos que una determinada vista necesita o en la que opera. Estos modelos de vista tienen interfaces y sus propios proveedores, "proveedores de vista del modelo", que entienden dónde obtener esta información de la capa de datos de la aplicación (por
15 ejemplo, que "modelos de datos" necesita un modelo de vista para solicitar datos y volver a formatear y/o combinar con el fin de agregar los datos necesarios).

[0026] En general, estas propiedades y operaciones son asíncronas, ya que ir a buscar los datos de una fuente externa (base de datos de red, servicios web, etc.) puede ser una operación lenta que de otro modo puede sostener la creación del
20 elemento que necesita los datos. Con las operaciones de búsqueda de datos asíncronas, un objeto de vista UI (o simplemente una "vista") se puede crear con la información que necesita buscar datos que la vista tiene la intención de mostrar al usuario, y la vista se actualiza cuando esos datos están disponibles. La propiedad de identificador "ID" en estos objetos no es asíncrona, sin embargo, debido a que la
25 propiedad ID es necesario para crear el objeto. La propiedad ID está destinada a contener los datos necesarios para identificar de forma única el objeto y para ir a buscar los datos necesarios, cuando se solicita. Esta propiedad, por lo tanto, no es asíncrona, debido a que una operación lenta de lo contrario puede sostener la

creación del objeto hasta que el servicio o similar, que proporciona el resto de los datos, se ha puesto en contacto.

[0027] Los proveedores, que son responsables de obtener los datos (haciendo una petición a alguna fuente externa y típicamente analizar el resultado), puede también almacenar en caché los resultados agarrándose de otros datos que regresa en la misma petición, por ejemplo, a través de la petición de múltiples propiedades a menudo que no requiere múltiples peticiones a la fuente de datos en red. Por ejemplo, cuando se le pide una interfaz Modelo de Usuario para el nombre de un usuario con un ID dado, el Proveedor de Usuario puede ponerse en contacto con un servicio que devuelve la información sobre el usuario solicitado (nombre, dirección, número de teléfono, etc.). Si los datos se almacenan en caché localmente, a continuación, una vista (o el número de vistas) que terminan solicitando diferentes propiedades del mismo proveedor (o la misma propiedad, más de una vez) no causan que las solicitudes lentas adicionales se realicen para traer los datos que ya han sido leídos. Los datos todavía se devuelven de forma asíncrona para que el objeto utilizando el modelo no necesite saber si el proveedor necesitó establecer una petición de red (generalmente lenta) o si el proveedor ya tenía los datos en el caché local.

[0028] El patrón asíncrono que se describe en el presente documento es útil debido a una serie de razones, incluyendo que el patrón desacopla de donde están los datos a cómo la aplicación obtiene los datos. Al actualizar a un nuevo formato de datos o nuevos servicios remotos, el proveedor puede tener que ser actualizado (o debe implementarse uno nuevo), pero ninguna otra entidad que utiliza el modelo se ve afectada o necesita ser actualizada.

[0029] Además, las interfaces de modelo proporcionan otra abstracción útil. Por ejemplo, un Menú de Modelo puede ser una interfaz genérica que define un modelo que puede proporcionar un título y una colección de artículos. Esta interfaz puede ser implementada por tanto por un Proveedor de Menú de Navegación (que sabe

cómo sacar el menú de navegación para una aplicación) y un Proveedor de Menú de Configuraciones (que sabe cómo construir una lista de los ajustes que se pueden personalizar en una aplicación). Los Proveedores de Modelo de Vista que operan en los datos del Menú de Modelo pueden trabajar con los datos para estos dos tipos de menús, sin saber de cuál se trata (o cómo se obtienen los datos cuando sea necesario).

[0030] De manera similar, una interfaz de vista de modelo puede ser implementada por más de un proveedor de vista del modelo. Por ejemplo, una Vista de Objeto de Menú (responsable de mostrar un elemento de menú a un usuario) puede operar en un Modelo de Vista de Objeto de Menú. El Modelo de Vista de Objeto de Menú puede ser una interfaz implementada por dos (o más) proveedores de vista del modelo (por ejemplo, uno para la visualización de un elemento de menú que une a otro submenú, y uno que se vincula a un activo en el nodo hoja de un árbol de menús). Aunque la forma de los datos es muy diferente (por ejemplo, El Modelo de Menú y el Modelo de Activo no pueden tener ningún campo en común), los proveedores de vista del modelo que implementan la interfaz Modelo de Vista de Objeto de Menú, cada uno sabe cómo tomar uno de estos modelos y asignar los datos de la forma que la Vista de Objeto de Menú lo espera.

[0031] La separación de vista/modelo de vista también es útil porque las vistas pueden ser accionadas por datos. Los modelos de vista representan lo que necesitan los datos para ser visualizados por la vista. La separación de vistas de los modelos de vista facilita la reutilización de componentes cambiando el modelo de vista subyacente, en lugar de sustituir toda la vista.

[0032] Las interfaces de modelo asíncronas permiten que los objetos se creen y a los datos a estar atados antes de que estén disponibles. El consumidor de los datos no necesita importarle (o saber) cómo se solicitarán/calcularan los datos. Este patrón mantiene la capacidad de respuesta de la aplicación, mientras que la solicitud de red lenta se completa.

[0033] Se debe entender que cualquiera de los ejemplos en el presente documento no son limitantes. Por ejemplo, aunque ciertos tipos de modelos y vistas se ejemplifican en el presente documento, la tecnología es capaz de trabajar con otros tipos de modelos y vistas. Como tal, la tecnología descrita en el presente documento no se limita a ninguna modalidad particular, aspectos, conceptos, estructuras, funcionalidades o ejemplos descritos en este documento. Más bien, cualquiera de las modalidades, aspectos, conceptos, estructuras, funcionalidades o ejemplos descritos en este documento no son limitantes, y la presente tecnología se puede utilizar de diversas maneras que proporcionan beneficios y ventajas en la computación, programas de aplicación y tecnología de interfaz de usuario utilizando modelos y vistas en general.

[0034] En general, para los propósitos de brevedad y simplicidad, como se usa en el presente documento un "modelo" se refiere normalmente al modelo de interfaz en relación con su aplicación asociada, que es el proveedor de modelo. Así, por ejemplo, como se usa en el presente documento una vista puede comunicarse con un "modelo de vista" (es decir, a través de la interfaz de modelo de vista), y a su vez el modelo de vista (por ejemplo, a través del código de proveedor) puede comunicarse con un modelo de datos (es decir, a través de la interfaz de modelo de datos). Tenga en cuenta que como se puede entender por su contexto, en algunas situaciones, un "modelo" puede referirse a una interfaz de objeto a un proveedor a una implementación del proveedor que implementa la interfaz y realiza el trabajo principal de ese objeto.

[0035] La figura 1 es un diagrama de bloques que muestra diversos componentes de ejemplo dirigidos hacia los modelos asíncronos. En una o más implementaciones, un modelo de vista 102 comprende una interfaz 104 que define la forma de los datos utilizados por una vista, tales como la vista 106. Un proveedor de modelo de vista 108 es una implementación del modelo de vista 102 que implementa la interfaz de modelo de vista 104. Tenga en cuenta que, en general, el

modelo de vista 102 básicamente especifica qué datos necesita su vista asociada, pero no se preocupa por la fuente de los datos. En su lugar, los datos 110 se utilizan para obtener los datos.

[0036] El modelo de datos 110 incluye una interfaz de modelo de datos 112 y el proveedor de modelo de datos 114 comprende una implementación del modelo de datos 110 (el proveedor del modelo de datos 114 implementa la interfaz del modelo de datos 112) que sabe cómo obtener los datos de una fuente de datos, como el servicio de web 116. Un modelo de vista 102 es, pues, diferente de un modelo (datos) 110 en el que la vista correspondiente 104 define la forma de datos necesaria por la vista, mientras que los servicios (tales como el servicio 112) determinan típicamente la forma de un modelo de datos 110.

[0037] A modo de ejemplo, como se representa generalmente en la figura 2A, para el mismo modelo de datos 220 (incluyendo la interfaz y el proveedor de datos), puede haber (por ejemplo, dos) diferentes modelos de vista (incluyendo interfaces y proveedores de vista) 222A y 222B para (por ejemplo, dos) puntos de vista diferentes 224A y 224B, respectivamente, que están cada uno unido al mismo modelo de datos 220. Cada uno de los modelos de vista 222A y 222B actúa como el "pegamento"/unión entre cada una de las vistas 222A y 222B y el modelo de datos 220, incluyendo la adaptación de los datos del modelo según sea necesario para el formato que cada vista 224A y 224B especifica.

[0038] En el ejemplo de la figura 2A, la vista 224A solicita el primer nombre y apellido para alguna persona, mientras que la vista 224B solicita (no necesariamente al mismo tiempo) el primer nombre, apellido y ocupación de esa persona. Tenga en cuenta que los puntos de vista 224A y 224B pueden no ser tan específicos en sus solicitudes de datos de vista específicos para los datos necesarios; por ejemplo, la vista 224A puede solicitar simplemente el "nombre" de una persona, con la vista del modelo 222A sabiendo que la vista 224A realmente quiere es el primer nombre y el apellido.

[0039] Como se describe aquí, los modelos de vista 226A y 226B (por ejemplo, sus proveedores) pueden romper solicitudes de datos en solicitudes más pequeñas asíncronas, por ejemplo, el nombre y luego el apellido para el modelo de vista 226A, y nombre, luego apellido, y luego ocupación para el modelo de vista 226B.

5 Alternativamente, es factible para cada vista romper sus propias peticiones. Esto se puede realizar por un número de razones, incluyendo que la solicitud de datos de vista especificado puede necesitar datos de dos (o más) modelos de datos, porque las solicitudes asíncronas más pequeñas pueden ser deseables (una solicitud para una gran cantidad de datos a la vez puede ser menos deseable que muchas
10 pequeñas solicitudes de ese mismo datos), y/o porque un modelo de datos puede estar diseñado para manejar el tiempo (u operar de forma más eficiente) de sólo una petición/respuesta en un momento (o algún pequeño número de solicitudes/respuestas a la vez).

[0040] El modelo de datos 220 conoce a través de la aplicación de su proveedor,
15 cómo obtener (al menos) esta información desde una fuente de datos apropiada, tal como un servicio de web 228, y en este ejemplo se busca un conjunto de datos 230 que comprende un primer nombre, apellido, dirección, cumpleaños y ocupación de la persona. Los modelos de vista 222A y 222B a través de sus respectivos proveedores de vista transforma cada formato de la información en datos
20 correspondientes, según sea necesario, por ejemplo, en los subconjuntos respectivos adecuados 232A y 232B del conjunto de datos 230 para su uso por las respectivas vistas 224A y 224B. En este momento las vistas 224A y 224B pueden usar los datos como se desee, por ejemplo, para visualizar los datos de acuerdo con un estilo apropiado, por ejemplo, tipo de letra, color de fuente, tamaño de fuente,
25 etc., dentro de un árbol de pantalla que se representa.

[0041] El modelo de datos 220 puede almacenar en caché cualquier información retornada (en conjunción con un identificador para facilitar la recuperación) en un caché de datos 234, por ejemplo, como el conjunto de datos almacenado en caché

230C con un identificador de memoria caché único X; (subconjuntos pueden ser almacenados en caché en lugar de o además de un conjunto de datos más grande).

El almacenamiento en caché sirve para varios propósitos, incluyendo almacenar en caché los datos sobrantes en previsión de su uso futuro. Por ejemplo, el primer nombre y apellido son susceptibles de ser solicitados, puede ocurrir en solicitudes separadas del mismo modelo de vista, como con el modelo de vista 222A, de esta manera almacenando en caché los dos juntos incluso si se solicita inicialmente sólo el primer nombre ahorra volver a la red para la siguiente solicitud por separado.

Del mismo modo, el almacenamiento en caché permite su reutilización por otra entidad solicitante; por ejemplo, si la vista 224B (a través de su modelo de vista 222B) realiza una solicitud o solicitudes del primer nombre, apellido y ocupación en un momento más tarde que la vista 224A (a través de su modelo de vista 222A), el modelo de datos 220 puede acceder a la información del caché de datos 234 en lugar de ir de nuevo al servicio web 228 (por lo general mucho más lento) para

recuperar de nuevo la información solicitada. Cabe señalar que modelo de vista/proveedor de modelo de vista también puede tener un caché de vista; por ejemplo, una vista de mosaico que comprende texto combinado con una imagen puede ser almacenada en caché para que la vista no tenga que ser recreada y reconstruida cada vez que se necesita la misma instancia de la vista de mosaico, incluso si los datos de texto y de imagen se almacenan en caché cada uno en la capa de datos.

[0042] Por lo tanto, teniendo el modelo de vista y la separación del modelo de datos, se proporcionan una serie de beneficios, incluyendo la capacidad de tener múltiples diferentes modelos de vista acoplados al mismo modelo de datos, y/o tener separadas las capas de vista de datos, lo que permite simplemente el cambio de un modelo de datos según sea necesario (por ejemplo, para trabajar con una nueva fuente de datos) sin afectar la capa de vista, y/o el uso de un nuevo modelo de vista por alguna nueva vista sin afectar la capa de datos. Cabe señalar, sin embargo, que es factible que algunas veces se tengan combinadas la estructura y la funcionalidad

del modelo de vista/el modelo de datos, por ejemplo, un único objeto de modelo (o incluso tener el proveedor (es) integrado en la vista), tal como para situaciones en las cuales una vista accede a los datos que nunca necesitará ninguna otra vista/modelo de vista.

5 [0043] Además de tener múltiples modelos de vista que se comunican con el mismo modelo de datos, dos o más modelos de vista pueden tener diferentes proveedores que implementan la misma interfaz de modelo de vista, como generalmente se ejemplifica en la figura 2B. Por ejemplo una vista W (así como posiblemente otras vistas) se comunica con un modelo de vista XY que tiene una interfaz de vista del
10 modelo X implementado por un proveedor de vista del modelo Y. La vista W también se comunica con un modelo de vista XZ que tiene una interfaz del modelo de vista X implementado por un proveedor de modelo de vista diferente Z. La vista W sólo necesita interactuar con (por ejemplo, genérico) una interfaz X que todavía puede obtener diversos datos de diferentes proveedores de modelo de vista.

15 [0044] La figura 3 resume los aspectos asíncronos del proceso para una sola vista 332 en diversas operaciones (solicitudes y respuestas) que se muestran a través de flechas etiquetadas. Cuando una vista 332 tiene un modelo de vista (interfaz 334 y proveedor 336) con datos que cambian (o con los datos que se establece inicialmente), la vista 332 necesita construir su nodo de árbol de visualización
20 basándose en los datos en el modelo de vista. La vista 332 hace esto preguntando de forma asíncrona a la interfaz de vista del modelo 334 por los datos, como se representa por la flecha etiquetada uno (1). Como se describió anteriormente, la interfaz de vista del modelo 334 es implementada por el proveedor de modelo de vista 336.

25 [0045] Para ciertos casos de vista, es factible para el proveedor del modelo de vista 336 tener datos de la vista almacenados en caché en un caché de vista local 338, y si es así, se comprueba la vista local de caché 338 (flecha (2) dos). En este ejemplo se debe tener en cuenta que los datos de vista no se almacenan en caché, por lo

que el proveedor del modelo de vista 336 traduce la solicitud en la solicitud correspondiente en la interfaz de modelo (datos) 340 (flecha tres (3)) con la que fue creada (por ejemplo, por una fábrica modelo como generalmente se describe a continuación con referencia a la figura 5). Por ejemplo, si los datos de la vista 332 necesitan datos llamados "texto", el proveedor del modelo de vista 336 puede saber para pedir a la interfaz de modelo apropiada 340 para un texto más específico como "título" en lugar de solicitar texto en general. Esto se puede hacer porque el modelo de datos entiende que la expresión "título" en su formato de solicitud de modelo de datos, y de hecho, puede no entender "texto", ya que puede ser diseñado para diferenciar entre los diversos elementos textuales como título, clasificación, reparto de personajes, fecha de lanzamiento de película, sinopsis, y así sucesivamente.

[0046] La interfaz de modelo de datos 340 se implementa mediante un proveedor de datos 342. El proveedor de datos 342 puede tener los datos en caché en un caché de datos local 344, y por lo tanto puede comprobar su caché de datos local 344 (flecha cuatro (4)). En este ejemplo, los datos no están (aún) almacenados en caché, por lo que el proveedor de datos 342 traduce la petición en una petición de web correspondiente (flecha cinco (5)). Las redes son asíncronas, por lo que la solicitud Web devuelve una promesa al proveedor 342 (flecha seis (6)), que se cumplirá una vez que la respuesta llegue.

[0047] El proveedor 342 no tiene todavía la respuesta, solamente una promesa de una, por lo que el proveedor también crea una promesa que va a transformar la respuesta web en los datos del modelo solicitado, una vez que se ha cumplido la promesa de las redes. La interfaz modelo 340 devuelve esta promesa creada al proveedor del modelo de vista 336 (flecha siete (7)).

[0048] El proveedor del modelo de vista 336 también aún no cuenta con los datos del modelo, solamente una promesa de estos, por lo que el proveedor del modelo de vista 336 crea una promesa que va a transformar los datos del modelo en datos del modelo de vista, una vez que se haya cumplido la promesa del modelo de datos.

La interfaz del modelo de vista 334 devuelve su promesa a la vista 332 (flecha ocho (8)); tenga en cuenta que esto se puede hacer antes de recibir la promesa del modelo de datos, o puede hacerse sólo una vez si la petición de una vista se divide en múltiples solicitudes de datos. La vista 332 todavía no puede crear su pantalla porque la vista 332 no tiene los datos de los modelos reales, por lo que la vista registra una devolución de llamada "después" o similar (por ejemplo, en un comunicado de promesa JavaScript®) para crear la pantalla de la vista cuando los datos llegan. Sin embargo, la vista sabe de la promesa, y puede realizar otras operaciones como realizar otras peticiones asíncronas a la espera de los datos.

10 [0049] En algún momento en el futuro, la respuesta web llega (flecha nueve (9)), y se cumple la promesa de las redes. El proveedor de datos analiza/traduce la respuesta al formato de modelo de datos/formato de información solicitado, y con estos datos analizados/traducidos cumple la promesa que regresó anteriormente (flecha diez (10)).

15 [0050] El proveedor del modelo de vista 336 es notificado de este modo del cumplimiento de la promesa del proveedor de datos, y obtiene los datos del modelo. El proveedor del modelo de vista 336 traduce esta de nuevo en los datos de modelo de vista especificado solicitados (por ejemplo, de "título" a "texto"), y cumple con la promesa que previamente devolvió, (por ejemplo, flecha once (11))

20 [0051] La promesa que se le dio a la vista 332 está así cumplida. La vista 332 obtiene los datos del modelo de vista, y el uso de los datos construye uno o más nodos del árbol de pantalla que se procesa para la representación en la salida para que el usuario lo pueda ver.

[0052] Volviendo a un ejemplo más específico, tal como se representa en la figura 4, consideremos una vista que muestra un nombre (nombre y apellido) como una cadena. Este punto de vista se llama Vista de Nombre 440, y su modelo de vista es un Modelo de Vista de Nombre 442. El Modelo de Vista de Nombre 442 comprende una interfaz que expone una propiedad "nombre" que devolverá una promesa para

25

una cadena. El Modelo de Vista de Nombre 442 es implementado por el Proveedor del Nombre de Usuario 444, que es un proveedor cuyo modelo es un Modelo de Usuario 446. El Modelo de Usuario 446 es una interfaz de modelo que expone propiedades para el primer nombre y apellido. El Proveedor de Usuario 448
5 implementa el Modelo de Usuario 446 y sabe cómo ir a un servicio de datos externo para obtener el primer nombre, apellido, dirección, y posiblemente otra información acerca de un usuario.

[0053] La figura 5 muestra un diálogo de secuencia de ejemplo que incluye el flujo de datos/pasos para una vista referida como Vista de Nombre 550 solicitando datos
10 de vista específicos (los datos que se necesitan) para datos de nombres ("obtener el nombre de") a partir de su modelo de vista (es decir, su interfaz y su implementación del proveedor colectivamente mostrados como Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553), y conseguir la cadena de nuevo de forma asíncrona. La figura 5 muestra un ejemplo donde un modelo de vista solicita
15 múltiples propiedades de un modelo de usuario.

[0054] Como puede verse, el Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553 rompe la solicitud en una pluralidad de solicitudes, es decir, una solicitud de "conseguir el primer nombre" y una solicitud de "obtener el apellido" en este ejemplo. En general, esto puede deberse a que el proveedor sabe que el
20 modelo de datos correspondiente espera peticiones en este formato, o para mantener las solicitudes pequeñas, y así sucesivamente.

[0055] La primera vez que se necesitan datos, a través de una petición "obtener el primer nombre", el modelo de datos (es decir, su interfaz y su aplicación de proveedor mostrados colectivamente en la figura 5 como Modelo de Usuario/
25 Proveedor de Usuario 555) comprueba su memoria caché, y determina que los datos necesarios no están almacenados en caché. La implementación del proveedor del Modelo de Usuario/Proveedor de Usuario 555 llama de este modo a un servicio externo 560 para obtener los datos y la implementación del proveedor

se encarga de la respuesta. Aunque no se muestra explícitamente en la figura 5, se entiende como se describe aquí que las solicitudes son asíncronas en este ejemplo, y por lo tanto las promesas se devuelven de manera apropiada, por ejemplo, debido a la demora en la recuperación de los datos.

- 5 [0056] En este ejemplo, cuando se cumple la solicitud del modelo de datos, la respuesta desde el servicio 560 devuelve un conjunto de datos que comprende más datos de los que se necesitaban, (nombre, apellido, dirección). Este conjunto de datos, o al menos parte del mismo, se almacenan en caché en la memoria caché del Modelo de Usuario/Proveedor de Usuario.
- 10 [0057] El primer nombre se devuelve al Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553 en respuesta a la solicitud de obtener el primer nombre, mediante el cual el Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553 luego solicita la siguiente propiedad deseada, el apellido, desde el Modelo de Usuario/Proveedor de Usuario 555. Esta vez, cuando el Modelo de Usuario/
- 15 Proveedor de Usuario 555 accede a su caché, los datos del apellido están presentes, por lo que la solicitud no tiene la necesidad de ir a la red (el servicio no tiene por qué ser contactado), ya que los datos en caché del apellido se devuelven al Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553. En este momento, el Modelo de Vista de Nombre/Proveedor de Vista de Nombre 553 tiene
- 20 los valores de la propiedad de nombre deseado del nombre y apellido, vuelve a ensamblar la información devuelta en el formato de datos "nombre" para que los datos devueltos estén en la forma adecuada para satisfacer la solicitud de los datos de vista especificados y devuelve el nombre de los datos a la Vista de Nombre 550 en respuesta a la solicitud original de obtener el nombre.
- 25 [0058] Volviendo a los detalles adicionales de un ejemplo de implementación, la figura 6 muestra un número de ejemplos de componentes de un entorno 600 incluyendo un sistema de interfaz de usuario (UI) que proporciona la estructura y funcionalidad anteriormente mencionadas. El sistema UI se puede utilizar, por

ejemplo, para cualquier aplicación, incluyendo una para la reproducción de la transmisión de vídeo, y puede ser ejecutada en la parte superior de cualquier plataforma de video/proveedor, proporcionando una capa de abstracción o similar que transforma la entrada y salida independiente de la plataforma a una entrada y salida apropiada para una plataforma en particular.

[0059] En una o más implementaciones, tales como en la figura 6, un navegador de objetos 601 es llamado por el código de inicio de aplicación 602 para desplazarse a un menú raíz (visualizado como un objeto Vista) correspondiente a la ubicación de inicio, por ejemplo, "Menú Raíz" (o algún valor de la variable representativa) se especifica como el identificador de fábrica (ID). Para obtener este menú, el navegador de objetos 601 llama a una fábrica de localización 604 para la ubicación 605 de este "Menú Raíz", (donde, en general, una fábrica es un objeto que sabe cómo devolver una instancia de un objeto solicitado en particular). Un anfitrión de vista 607 se crea a través de una Fábrica de Anfitrión de Vista 608 (llamada por la ubicación 605) para alojar un menú de visualización 609, donde el menú de visualización 609 a su vez es creado por una Fábrica de Vista 610. Por lo tanto, el anfitrión de vista 607 llama a la fábrica de vista 610 para devolver la vista solicitada 609, aplicando típicamente un estilo a la vista, (por ejemplo, esquema de color, tamaño del texto, animación y así sucesivamente), como se especifica por un diseñador de la vista 609 de entre un conjunto de estilos 611, y que también puede estar basado en el ID de fábrica para la vista solicitada en particular, que en este ejemplo es un menú (o una vista de ítems). Tenga en cuenta que el conjunto de estilos puede ser plano para cada vista en el tiempo de carga, en lugar de cambiar dinámicamente los datos de estilo asociados de una vista en el tiempo de ejecución.

[0060] En una o más implementaciones, los estilos son utilizados por la fábrica de vista 610 para inicializar las propiedades de los objetos creados. Los estilos están asociados con una ID de fábrica, y se pueden configurar por el cliente. El anfitrión de vista 607, si es visualizado, también puede tener un estilo aplicado de la misma

manera desde el conjunto de estilo 611 a través de la Fábrica de Anfitrión de Vista 608.

[0061] En las situaciones típicas, la vista 609 puede necesitar datos relacionados con vistas, que a su vez pueden ser dependientes en los datos de un modelo (datos). Por lo tanto, una fábrica del modelo de vista 620, proveedor de modelo de vista 621 que implementa una interfaz del modelo de vista 622 pueden ser objetos llamados por otros objetos del sistema como se representa en la figura 6; en general, un proveedor es un objeto que sabe cómo recuperar (y si es necesario) formatear la información correspondiente a un modelo. Del mismo modo, una fábrica de modelo (datos) 624 crea un proveedor (datos) 625 con su interfaz de modelo de datos implementado 626. El proveedor 625 recupera los datos necesarios, por ejemplo, de la plataforma de red 630 que se comunica con uno o más servicios 632; (Tenga en cuenta que el proveedor de modelo de vista 621 convierte los datos del formato de proveedor de datos 625 a un formato que necesite la vista 609).

[0062] A modo de ejemplo, la vista "Menú Raíz" puede contener otras vistas, comprendiendo cada una un botón u otro componente interactivo que cuando es seleccionado manda al usuario a un menú diferente. Cada uno de estos botones (vistas) pueden estar asociados con una ubicación de navegación (otra vista) a la cual el sistema se desplaza al pulsar dicho botón. A su vez, esas otras vistas tienen un modelo de datos asociado que especifica sus datos, que también pueden incluir botones asociados a otras ubicaciones de navegación.

[0063] En resumen, para navegar a una ubicación, una fábrica de localización 604 correlaciona el nombre de un lugar navegable en un tipo concreto de ubicación 605. La fábrica de localización 604 permite a cada cliente (un programa/plataforma usando el sistema de fábrica) para ser configurado para tener diferentes ubicaciones concretas. La fábrica de anfitrión de vista 608 es responsable de crear un anfitrión de vista 607 desde una fábrica de anfitrión de vista, y aplicando un estilo de anfitrión de vista apropiado. Para una localización dada, puede haber múltiples anfitriones

de vista, por ejemplo, porque a pesar de que los diferentes clientes pueden compartir el mismo lugar/datos (ubicación), pueden parecer diferentes visualmente (por ejemplo, un diseño de diez pies frente a un diseño de tableta son típicamente diferentes entre sí). Además, para un único cliente, ciertas condiciones de tiempo de ejecución pueden dictar diferentes vistas del mismo lugar (por ejemplo, el modo de referencia frente al modo completo frente al modo de relleno que son posibles modos que determinan cómo aparece una vista).

[0064] La fábrica de vista 610 es responsable de la creación de una vista 609 de una vista de ID de fábrica, y aplicando un estilo de vista adecuado. Tenga en cuenta que una diferencia entre una ID de fábrica y un tipo de vista es que un solo tipo (por ejemplo, un objeto contenedor de Vista de Ítems) puede ser utilizado por múltiples configuraciones Id de fábrica, por ejemplo, un contenedor de lista puede contener diferentes menús que tienen cada uno un ID de fábrica diferente, por ejemplo. Una diferencia entre un ID de fábrica y una instancia es que el sistema puede acabar con varias instancias para el mismo ID de fábrica. Las configuraciones de la fábrica de vista son básicamente configuraciones prefabricadas de vista.

[0065] La fábrica del modelo de vista 620 es responsable de crear una instancia de un proveedor del modelo de vista 621 (que implementa una interfaz del modelo de vista 622) para un ID de fábrica, dado un modelo. La fábrica del modelo de vista 620 decide qué implementación de interfaz del modelo de vista utilizar (que proveedor de modelo de vista 621).

[0066] La fábrica de modelo (datos) 624 es responsable de crear una instancia de un proveedor 625 (que implementa una interfaz de modelo) para un ID de fábrica, dado un ID de datos. La fábrica de modelo 624 decide qué implementación de un modelo de interfaz a utilizar (el proveedor de modelo 625). Por ejemplo, un proveedor de datos sabe cómo recuperar los datos solicitados al servicio de red 632.

[0067] Un modelo puede considerar una interfaz que define la forma de los datos utilizados por el programa de aplicación, y un proveedor es una implementación de un modelo. Un modelo describe lo que los datos son, no de dónde vienen. Los modelos pueden tener identificadores (IDs), que se utilizan para el almacenamiento en caché. La creación de un modelo es sincrónico, pero el acceso a él (aparte de su ID) es asíncrono; por ejemplo, promesas de regreso de propiedades del modelo. Un proveedor de datos sabe dónde obtener los datos que define un modelo, y adapta los datos que obtiene de sus fuentes originales al formato que especifica el modelo. No puede haber varios proveedores diferentes para el mismo modelo, si hay varios lugares para obtener los datos. Es común que los proveedores hagan llamadas a la pila de plataforma HTTP para solicitar los datos de un servicio web, por ejemplo.

[0068] Con respecto a las vistas, una vista es una unidad primaria de componibles UI, y una vista puede tener uno o más puntos de vista de niño. Una vista tiene un Nodo de Visualización asociado (árbol) 640 para la visualización de la vista, (como también se representa en general en la figura 6). Los subnodos normalmente se crean sobre la base de los datos del modelo de vista. La vista normalmente crea/pasa un modelo de vista derivada de su propio modelo de vista a las vistas de niño. Un punto de vista, si es invocable, se encarga de la entrada del usuario que es específica para la vista. Esto a veces se traduce en una navegación a través del Enrutador de Navegación 650, por ejemplo, a una vista diferente.

[0069] De esta manera, una vista contenedora, tal como un menú que contiene botones, puede navegar a través de una selección de botón a otros lugares. La vista está construida con vistas secundarias (tales como botones o azulejos), donde cada vista se convierte en un nodo de visualización 640 en un árbol de nodos de visualización. El árbol se representa a un usuario 641 (a través de la plataforma de representación 642), mediante el cual el usuario puede interactuar con la vista 609 a través de la entrada de plataforma 644, por ejemplo, a un dispositivo de

botón/proveedor de botón 646, 647 que se maneja por un gestor de entrada 648. La interacción con la vista 609 resulta en un enrutador de navegación 650 que se invoca para navegar a una nueva ubicación, por ejemplo, otro menú que comprende una vista de contenedor con un conjunto de vistas secundarias con las que un usuario 641 puede interactuar, y así sucesivamente, por ejemplo, para proporcionar la entrada 644 y para ver una representación 642 de los nodos de visualización 640.

[0070] Nótese que aunque la figura 6 muestra un dispositivo de botón 646, se utilizan otros tipos de entrada además de la entrada de botón. En una o más implementaciones, esto incluye la entrada de puntero (por ejemplo, a través de un ratón o un ratón táctil) y/o la entrada de comandos (por ejemplo, a través del habla o las teclas del teclado avanzadas/combinaciones de teclas que corresponden a los comandos en lugar de botones). En general, la entrada de plataforma 644 se pasa al nivel independiente de la plataforma y llega a una vista a través del gestor de entrada 648 y un alcance de entrada asociado de entrada 649. La entrada de botón se encamina a una visión que tiene enfoque, la entrada de puntero se encamina a una vista basada en pruebas de éxito, y entrada de comando que se encamina a una vista (o conjunto de puntos de vista) que cumple con los criterios de filtrado con respecto a la manipulación del tipo de comando recibido y/o cómo se recibió ese comando (por ejemplo, por el habla frente a una combinación de teclas del teclado).

[0071] También se muestra en la figura 6 la capacidad para la entrada automatizada, que se muestra a través de la automatización 671, por ejemplo, sobre la base de un árbol de automatización 672 o similar, para actuar como un dispositivo de botón virtual 673 (u otro dispositivo de entrada virtual). Esto se puede utilizar en las pruebas, solución de problemas y así sucesivamente, por ejemplo, incluyendo a través de la operación remota.

[0072] Entre otras funciones, el navegador 601 también mantiene navegación de las migas (una pila de navegación de ubicaciones) para que los usuarios pueden navegar a las ubicaciones anteriores. Más particularmente, el navegador 604 en

conjunción con la fábrica de localización 606 crea lugares en la navegación, y cuando se navega fuera de un lugar, el navegador 604 puede guardar el estado de la ubicación en una ruta de navegación.

[0073] El Enrutador de Navegación 650 es responsable de decidir una ubicación a la cual navegar, dado un modelo y un poco de contexto adicional. Es común que un botón este asociado con algunos datos, y cuando se hace clic en el botón, el sistema necesita navegar hasta una ubicación que está determinada por el tipo de datos con los que se asocia el botón (ligado). Esto consolida la lógica. Tal navegación basada en datos proporciona numerosos beneficios, incluyendo que un cambio en los datos resulta en un cambio en el comportamiento del programa, sin necesidad de cambiar el código y/o enlaces en una página, por ejemplo.

[0074] Como se describe aquí, una ubicación 605 representa una ubicación navegable (y potencialmente enlazable a lo profundo) en el código de programa (por ejemplo, aplicación). La ubicación es responsable de obtener los datos asociados con esta ubicación, teniendo en cuenta algunos argumentos de navegación, y la conexión de la ubicación a un Anfitrión de Vista 607. Tenga en cuenta que en una implementación, la ubicación no sabe nada acerca de la vista 609, sólo los datos, es decir, los modelos. En general, una ubicación es más portátil que los Anfitriones de Vista; por ejemplo, diferentes diseños se pueden construir fuera de los mismos datos.

[0075] El Anfitrión de Vista 607 es responsable de crear las Vistas que visualizan el modelo proporcionado por un punto. El Anfitrión de Vista 607 registra las vistas que se pueden crear o adquirir a través de transferencia. El Anfitrión de Vista 607 también tiene que ser capaz de crear una vista registrada si no es transferida por otro Anfitrión de Vista anterior, y tiene que ser capaz de identificar y traspasar una vista si ya fue creada por el Anfitrión de Vista 607 anterior. El Anfitrión de Vista 607 sabe acerca de las relaciones entre vistas, y por lo tanto es capaz de manejar datos de entrada que cruzan el límite de vista de nivel superior.

[0076] La figura 7 es un diagrama de flujo que muestra varios pasos de ejemplo a partir de la perspectiva de un objeto de vista. Las figuras 8 y 9 comprenden un diagrama de flujo que muestra varios pasos de ejemplo a partir de la perspectiva de un modelo de vista. La figura 10 es un diagrama de flujo que muestra varios pasos de ejemplo desde la perspectiva de un modelo de datos. Tenga en cuenta que las figuras 7-10 suponen que los datos se devuelven para que se cumplan las promesas, y no alude errores explícitamente, sin embargo, uno o más mecanismos de control de errores pueden ser incorporados en el programa de aplicación y es sencillo para los objetos hacerle frente a las promesas rechazadas.

[0077] El paso 702 de la figura 7 representa la vista que recibe una petición para extraer alguna salida, por ejemplo, para mostrar el nodo (s) de un árbol de pantalla. Como se entiende, la petición puede ser de uno de los padres, por ejemplo, una vista superior o, posiblemente, el anfitrión de vista. En el ejemplo de la figura 7, la vista necesita de datos para extraer, y por lo tanto realiza una solicitud al modelo de vista asociado a la vista en el paso 704.

[0078] En este ejemplo, la vista recibe una promesa del modelo de vista del paso 706, y en el paso 708 registra una devolución de llamada (por ejemplo, un JavaScript® "y luego" la devolución de llamada o similar") cuando los datos se encuentren disponibles. Mientras se espera los datos, la vista puede seguir haciendo otro trabajo como se representa por la etapa 710, lo que puede incluir la fabricación de otras solicitudes de datos a uno o más de otros proveedores de datos (por ejemplo, la primera petición puede ser para texto de un modelo de vista, y una segunda solicitud puede ser para una imagen desde otro modelo de vista, o, posiblemente, una segunda solicitud para el mismo modelo). Otro ejemplo de trabajo puede ser para instruir a una vista secundaria para prepararse para dibujar, de modo que el nodo secundario puede comenzar la recuperación de los datos que la vista secundaria necesitará cuando sea el momento en el que la vista secundaria vaya a dibujar. Hay que tener en cuenta que normalmente una vista secundaria en

realidad no dibuja hasta que el padre atrae de manera que el secundario dibuja en la parte superior de la matriz en orden z, sin embargo, es factible tener una vista pre-secundaria que haga un poco de resaltado o similar antes que la vista padre dibuje para que la vista pre-secundaria dibuje por debajo de la vista padre.

5 [0079] Otro tipo de trabajo que un objeto de vista puede llevar a cabo es dibujar como un "marcador de posición" en espera de (por lo menos algunos) sus datos. Por ejemplo, considere a un usuario desplazarse rápidamente a través de un menú que contiene un gran número de artículos (por ejemplo, 1,000), cada uno apoyado por los datos de red, donde 20 elementos aparecen a la vez. Una vista que
10 representa un artículo puede dibujarse a sí mismo como un marcador de posición (por ejemplo, un rectángulo en blanco o en su mayoría en blanco si algunos datos están disponibles), sin embargo tener toda o incluso cualquiera de los datos necesarios para visualizarse completamente en sí. Esto permite a un usuario continuar desplazándose rápidamente para llegar a un cierto punto de parada (por
15 ejemplo, los artículos 481 a través de 500) sin esperar a que cada conjunto completo de datos sean recibidos para cada vista del elemento; una vez detenida, esas vistas de elementos se visualizarán totalmente una vez que los datos se reciben. Las peticiones asíncronas en espera pueden ser canceladas si ya no se necesitan los datos, por ejemplo, debido a que el usuario se ha desplazado más allá de esos
20 artículos (aunque al menos algunos de dichos datos en su lugar pueden ser recibidos y se almacenan en caché en previsión de una necesidad más tarde de los mismos).

[0080] El paso 712 representa la espera de la devolución de llamada, realizando cualquier otro trabajo a través del paso 710 hasta que los datos estén disponibles.

25 Debido a que la petición/devolución de llamada son operaciones asíncronas, la vista no bloquea (incluso en un programa de aplicación de un solo subproceso) para que la vista y otras vistas y sus objetos relacionados puedan realizar un trabajo.

[0081] En algún momento más tarde, en el paso 714 la vista recibe los datos de vista-especificados del modelo de vista, que cumple la promesa (paso 706). El paso 716 representa la vista sacando su visualización de estilo de los datos al nodo (s) de visualización para la representación.

5 [0082] Los pasos de modelo de vista de ejemplo de la figura 8 empiezan en el paso 802 donde el modelo de vista recibe la solicitud desde la vista para los datos de vista especificados. El paso 804 devuelve una promesa del modelo de vista a la vista. Hay que tener en cuenta que (como se describe anteriormente con referencia a la figura 5) es factible para este paso que se produzca más tarde, por ejemplo, cuando el modelo de vista recibe una promesa del modelo de datos, sin embargo ya que sólo una petición fue recibida de la vista, el modelo de vista devuelve sólo una promesa a la vista (a pesar de que el modelo de vista puede romper la solicitud y enviar varias solicitudes de propiedad sobre el modelo de datos y recibir múltiples promesas a cambio). Otra alternativa es la de devolver múltiples promesas asociadas con una única petición de una vista, con la vista haciendo caso omiso de todas las promesas con excepción de la primera promesa. También es factible que el modelo de vista pueda tener su propio caché; si es así, los datos solicitados pueden estar situados ahí mismo, sin embargo, para los propósitos de este ejemplo se considera que el modelo de vista no tiene una memoria caché, o que los datos de la vista especificada solicitada no se almacenan en caché y por tanto el modelo de datos debe ser llamado para obtener la información correspondiente a los datos de vista-especificados que son solicitados.

[0083] El paso 806 representa el modelo de vista traduciendo la petición/rompiendo la solicitud en el formato compatible con el modelo de datos; (esto fue ejemplificado anteriormente como la traducción de "texto" a "título" en un ejemplo anterior, y partiendo el "nombre" en las solicitudes de nombre y apellido en otro ejemplo más arriba). Los pasos 808 y 810 representan el envío de la primera solicitud al modelo de datos para la primera parte de la información; (obsérvese que puede haber sólo

una parte, pero como se ejemplifica con respecto a la figura 5, una solicitud de datos de vista-especificados puede estar dividida en múltiples solicitudes de partes de la información, por ejemplo, el "nombre" fue particionado hasta en dos partes de la información necesaria, correspondientes a las peticiones de "Nombre" y "Apellido").

- 5 En efecto, una solicitud puede ser dividida en múltiples partes para enviar a dos o más proveedores de datos diferentes.

[0084] El paso 812 representa la recepción de una promesa del modelo de datos en respuesta a la solicitud de información. Los pasos 814 y 816 registran y esperan a la devolución de llamada, respectivamente. En el paso 818 se cumple la promesa de la información solicitada a partir del modelo de datos.

[0085] Los pasos 820 y 822 repiten el proceso para cada parte de la información necesaria, hasta que no permanecen partes de información necesaria para completar la solicitud de datos de vista-especificados. Cuando esto ocurre, el proceso continúa al paso 902 de la figura 9.

- 15 [0086] El paso 902 representa la traducción y/o el montaje de la información del modelo de datos en el formato necesario por la vista, por ejemplo, nombre y apellido en una sola respuesta "nombre", o "título" de nuevo a "texto" para continuar con los ejemplos anteriores. El paso 904 devuelve la respuesta con los datos de vista especificados para cumplir la promesa del paso 804 de la figura 8, que corresponde al paso 714 de la figura 7 desde la perspectiva de la vista.

[0087] La figura 10 representa operaciones de ejemplo del modelo de datos, comenzando en el paso 1002 donde el modelo de datos recibe una solicitud del modelo de vista para devolver información. El paso 1004 devuelve una promesa para el modelo de vista, porque la solicitud es asíncrona. Tenga en cuenta que esta promesa se puede devolver después, tal como al recibir una promesa de la fuente de datos.

[0088] El paso 1006 representa el modelo de datos en busca de una copia en caché del conjunto de datos o similar, que contiene la información deseada, por ejemplo,

mediante un ID de caché-único apropiado. Si se encuentra, el paso 1008 obtiene la información del caché, después del cual se devuelve al paso 1022. De lo contrario, el modelo de datos (la implementación del proveedor) necesita para obtener los datos a través de una fuente externa de datos, por ejemplo, un servicio, por el cual el paso 1010 hace la solicitud.

[0089] El servicio u otra fuente de datos pueden devolver una promesa que el modelo de datos recibe en el paso 1012. También es factible para tal promesa ser implícitamente creada por el modelo de datos, (por ejemplo, una fuente de datos puede ser alguna entidad que no crea promesas sino simplemente devuelve datos), debido a que el modelo de datos sabe que está haciendo una solicitud de red para un conjunto de datos, que es relativamente lento. En cualquier caso, los pasos 1014 y 1016 esperan el conjunto de datos de la fuente de datos, que puede ser una devolución de llamada formal o algún otro tipo de comunicación, sin embargo, la solicitud es asíncrona / no bloquea mediante el cual otros trabajos de aplicación pueden ser realizados.

[0090] El paso 1018 recibe el conjunto de datos, y se cumple la promesa (explícita o implícita). El paso 1020 almacena en caché el conjunto de datos con un ID apropiado. La información solicitada desde el conjunto de datos se devuelve al modelo de vista en el paso 1022; nótese que esto sólo puede ser parte de la información necesaria, y el modelo de vista puede volver a solicitar otra parte de la información en una llamada posterior, con lo cual los pasos de la figura 10 se repiten (con el conjunto de datos probablemente almacenado en caché con la información solicitada en la petición subsiguiente o las solicitudes).

[0091] Hay que señalar que a pesar de todo sigue asíncrona en una o más implementaciones, el modelo de datos (o de manera similar al modelo de vista) no tienen por qué regresar una promesa si la información solicitada se almacena en caché y es más eficiente para volver directamente la información, en lugar de devolver una promesa seguida de la información. Por ejemplo, si la información se

encuentra en una caché de memoria RAM, entonces es probablemente más eficiente devolver la información del caché de memoria RAM directamente, en lugar de devolver una promesa, ir a buscar la información (relativamente muy rápido) del caché de memoria RAM, y regresar la información para cumplir con la promesa; por tanto, el paso 1004 de la figura 10 puede venir después del paso 1006 en una situación tal que una promesa no se devuelva para los datos en caché, con el paso 812 de la figura 8 que recibe los datos almacenados en caché en lugar de una promesa y saltar al paso 820. Además, es factible que la promesa del modelo de vista a la vista (paso 804 de la figura 8) sea enviada solamente si el modelo de vista recibe una promesa a partir del modelo de datos en lugar de la información solicitada/parte de la información.

[0092] Sin embargo, hay varios tipos de medios de memoria caché, por lo que no siempre puede ser más eficiente saltar el mecanismo de promesa/cumplimiento. Por ejemplo, si la memoria caché del modelo de datos está en la memoria virtual, esta puede ser almacenado en caché, pero respaldada por una memoria de unidad flash o disco duro en lugar de la memoria RAM cuando sea necesario, mediante el cual la recuperación puede ser lo suficientemente lenta como para afectar negativamente al proceso de programa de aplicación general en relación con el uso de un mecanismo de promesa/cumplimiento o similares. Si se intenta eludir el mecanismo de promesa/cumplimiento con acceso directo a la información, es necesario tener en consideración de que tan eficiente en realidad es el acceso a la información almacenada.

[0093] Como puede verse, están descritos los modelos asíncronos para un programa de aplicación/sistema UI. Una capa de vista es distinta de una capa de datos, con vistas que tienen modelos de vistas asociados con las vistas acopladas a modelos de datos asociados con las fuentes de datos, proporcionando una serie de beneficios. Las operaciones asíncronas permiten un modelo que se creó inicialmente con todo, ninguno, o algunos de sus datos, mientras que las vistas se

pueden crear y funcionar, en gran medida, incluso cuando esos puntos de vista dependen de los datos que un programa de aplicación no tiene todavía.

[0094] Uno o más aspectos están dirigidos a recibir, en un modelo de vista, una petición de una vista para ver datos especificados, y en el modelo de vista, haciendo

5 una o más solicitudes asíncronas a un modelo de datos para la información correspondiente a los datos de vista especificados. Descrito en el presente documento está el devolver una promesa del modelo de vista a la vista en respuesta a la solicitud de los datos de vista especificados, y la obtención de un conjunto de datos que incluyen la información en el modelo de datos. La información se
10 proporciona desde el modelo de datos para el modelo de vista en respuesta a una o más peticiones asincrónicas, y los datos de vista especificados vuelven, basándose en la información devuelta al modelo de vista, desde el modelo de vista a la vista para cumplir la promesa.

[0095] Una promesa del modelo de datos al modelo de vista puede ser devuelta en
15 respuesta a la solicitud de información. Hay que tener en cuenta que la vista puede ser creada a través de una fábrica de vista, el modelo de vista puede ser creado a través de una fábrica del modelo de vista y el modelo de datos puede ser creado a través de una fábrica de modelo de datos.

[0096] La obtención del conjunto de datos puede incluir la comunicación con una
20 fuente de datos, devolviendo una promesa del modelo de datos para el modelo de vista, recibir el conjunto de datos de la fuente de datos, y devolver al menos parte de la información del modelo de datos para el modelo de vista para cumplir la promesa. Una promesa de la fuente de datos puede ser recibida en el modelo de datos. La obtención del conjunto de datos que incluye la información en el modelo
25 de datos puede incluir el acceso a la base de datos en una memoria caché acoplada al modelo de datos.

[0097] Al menos parte del conjunto de datos recibidos de la fuente de datos puede ser almacenado en caché en una memoria caché acoplada al modelo de datos. Al

recibir otra solicitud en el modelo de datos para información solicitada, el modelo de datos puede determinar que la información solicitada está en la caché, y devuelve la información de la memoria caché en respuesta a la otra petición.

[0098] Haciendo uno o más peticiones asíncronas al modelo de datos para información correspondiente a los datos de vista especificados, puede incluir hacer al menos dos peticiones separadas para diferentes partes de la información. En el modelo de datos al menos una parte del conjunto de datos puede ser almacenada en caché en un modelo de datos de caché, por lo que la información del modelo de datos para el modelo de vista en respuesta a una o más peticiones asíncronas puede incluir proporcionar una primera parte de la información a partir de la fuente de datos en respuesta a una petición, y proporcionando otra parte de la información de caché en respuesta a otra petición.

[0099] Uno o más aspectos se dirigen hacia un modelo de vista que comprende una interfaz del modelo de vista y un proveedor del modelo de vista, el modelo de vista configurado para ser acoplado a un objeto de vista a través de la interfaz del modelo de vista, junto con un modelo de datos que comprende una interfaz de modelo de datos y un proveedor de modelo de datos. El modelo de vista está configurado para ser acoplado al modelo de datos a través de la interfaz de modelo de datos, y está configurado para recibir una solicitud de datos de vista especificados del objeto de vista. El proveedor del modelo de vista solicita de forma asíncrona la información correspondiente a los datos de vista especificados desde el modelo de datos, con el proveedor de modelo de datos configurado para obtener un conjunto de datos que incluye la información de una fuente de datos y devolver la información al modelo de vista. El modelo de vista devuelve una respuesta que comprende los datos de vista especificados al objeto de vista en base a la información.

[00100] El proveedor de modelo de vista puede romper la solicitud de los datos de vista especificados en una pluralidad de solicitudes asíncronas al modelo de datos para las partes de la información. El proveedor del modelo de vista puede volver a

montar las partes de la información recibida del modelo de datos en la respuesta que comprende los datos de vista especificados al objeto de vista. El proveedor del modelo de vista puede traducir la información recibida del modelo de datos en la respuesta a un formato de datos correspondiente a los datos de vista especificados para volver al objeto de vista. El modelo de vista puede devolver una promesa para el objeto de vista, y más tarde devolver la respuesta que comprende los datos de vista especificados al objeto de vista para cumplir la promesa.

[00101] La fuente de datos puede comprender un servicio web. La fuente de datos puede comprender una memoria caché, y el modelo de datos puede recuperar la información del caché.

[00102] Uno o más aspectos están dirigidos a recibir, en un modelo de vista, una petición de una vista para datos de vista especificados, devolver una primera promesa del modelo de vista a la vista y haciendo una solicitud asincrónica a un modelo de datos para obtener información correspondiente a los datos de vista especificados. Se recibe una segunda promesa del modelo de datos para el modelo de vista. El modelo de datos realiza una solicitud asíncrona a una fuente de datos para unos datos correspondiente a la información, y obtiene el conjunto de datos en respuesta a la solicitud a la fuente de datos en el modelo de datos, donde el conjunto de datos incluye la información. Descrito en el presente documento está el regreso de la información del modelo de datos al modelo de vista para cumplir la segunda promesa, y la devolución de los datos de vista especificados, basándose en la información devuelta al modelo de vista, desde el modelo de vista a la vista para cumplir la primer promesa. Una tercera promesa de la fuente de datos al modelo de datos puede ser recibida, con lo que la obtención del conjunto de datos en respuesta a la solicitud a la fuente de datos en el modelo de datos cumple la tercera promesa.

[00103] En el modelo de vista, la información del modelo de datos puede ser transformada en un formato correspondiente a los datos de vista especificados. La solicitud asíncrona al modelo de datos para la información correspondiente a los

datos de vista especificados puede ser parte de una pluralidad de solicitudes, y al menos otra solicitud asíncrona puede ser hecha para el modelo de datos para la otra información correspondiente a los datos de vista especificados.

5 DISPOSITIVO DE COMPUTO DE EJEMPLO

[00104] Las técnicas descritas en el presente documento pueden aplicarse a cualquier dispositivo o conjunto de dispositivos (máquinas) capaces de ejecutar programas y procesos. Se puede entender, por lo tanto, que los ordenadores personales, ordenadores portátiles, ordenadores de mano, portátiles y otros dispositivos informáticos y objetos de todo tipo, incluyendo teléfonos celulares, tabletas/ordenadores tipo pizarra, consolas de juegos /entretenimiento y similares se contemplan para su uso en relación con diversas implementaciones que incluyen las ejemplificadas en el presente documento. En consecuencia, el mecanismo informático de propósito general que se describe a continuación en la figura 11 es sólo un ejemplo de un dispositivo de computación.

[00105] Las implementaciones en parte pueden ser implementadas a través de un sistema operativo, para su uso por un desarrollador de servicios para un dispositivo u objeto, y/o incluidos dentro del software de aplicación que opera para realizar uno o más aspectos funcionales de las diversas implementaciones descritas en este documento. El software puede describirse en el contexto general de las instrucciones ejecutables por el ordenador, tales como módulos de programa, ejecución por uno o más ordenadores, tales como estaciones de trabajo, servidores u otros dispositivos. Los expertos en la técnica apreciarán que los sistemas de computación tienen una variedad de configuraciones y protocolos que pueden utilizarse para comunicar datos, y por lo tanto, no se considera ninguna configuración o protocolo particular limitante.

[00106] La figura 11 ilustra así un ejemplo de un entorno de sistema informático adecuado 1100 en el que se pueden implementar uno o más aspectos de las

implementaciones descritas en este documento, aunque, como se dejó claro anteriormente, el ambiente de sistema de computación 1100 es sólo un ejemplo de un entorno informático adecuado y no se pretende sugerir ninguna limitación en cuanto al alcance de uso o funcionalidad. Además, el ambiente de sistema informático 1100 no está destinado a ser interpretado como que tiene ninguna dependencia relativa a cualquiera o a una combinación de componentes ilustrados en el ambiente de sistema informático de ejemplo 1100.

[00107] Con referencia a la figura 11, un dispositivo de ejemplo para la implementación de una o más implementaciones incluye un dispositivo informático de propósito general en la forma de un ordenador 1110. Los componentes del ordenador 1110 puede incluir, pero no se limitan a, una unidad de procesamiento 1120, una memoria de sistema 1130, y un bus de sistema 1122 que acopla varios componentes del sistema, incluyendo la memoria del sistema a la unidad de procesamiento 1120.

[00108] El ordenador 1110 incluye típicamente una variedad de medios legibles por máquina (por ejemplo, ordenador) y puede ser cualquier medio disponible al que se pueda acceder mediante una máquina tal como el ordenador 1110. La memoria del sistema 1130 puede incluir medios de almacenamiento de ordenador en forma de memoria volátil y/o no volátil tal como la memoria de sólo lectura (ROM) y/o la memoria de acceso aleatorio (RAM), y los medios de disco duro, medios de almacenamiento óptico, medios flash, y así sucesivamente; como se usa en el presente documento, los medios de almacenamiento legibles por máquina/ordenador almacenan datos que no incluyen señales transitorias, (aunque pueden ser otros tipos de medios legibles por máquina/ordenador que no son medios de almacenamiento). A modo de ejemplo, y no de limitación, la memoria del sistema 1130 también puede incluir un sistema operativo, programas de aplicación, otros módulos de programa, y datos de programa.

[00109] Un usuario puede introducir comandos e información en el ordenador 1110 a través de uno o más dispositivos de entrada 1140. Un monitor u otro tipo de dispositivo de visualización también está conectado al bus del sistema 1122 a través de una interfaz, como la interfaz de salida 1150. Además de un monitor, los ordenadores también pueden incluir otros dispositivos periféricos de salida tales como altavoces y una impresora, que pueden ser conectados a través de la interfaz de salida 1150.

[00110] El ordenador 1110 puede operar en un entorno de red o distribuido utilizando conexiones lógicas con uno o más de otros ordenadores remotos, tales como el equipo remoto 1170. El ordenador remoto 1170 puede ser un ordenador personal, un servidor, un enrutador, un PC en red, un dispositivo par u otro nodo de red común, o cualquier otro dispositivo de transmisión o consumo de medios remotos, y pueden incluir cualquiera o todos los elementos descritos anteriormente con respecto al ordenador 1110. Las conexiones lógicas representadas en la figura 11 incluyen una red 1172, tal como una red de área local (LAN) o una red de área amplia (WAN), pero también pueden incluir otras redes/buses. Tales entornos de red son comunes en los hogares, oficinas, redes de ordenadores de empresa, intranets e Internet.

[00111] Como se mencionó anteriormente, mientras que las implementaciones de ejemplo se han descrito en relación con diversos dispositivos informáticos y arquitecturas de red, los conceptos subyacentes pueden aplicarse a cualquier sistema de red y cualquier dispositivo informático o sistema donde es deseable implementar dicha tecnología.

[00112] Además, hay múltiples formas de implementar la misma funcionalidad o similar, por ejemplo, un API apropiado, caja de herramientas, código controlador, sistema operativo, control, objeto de software autónomo o descargable, etc., lo que permite a las aplicaciones y servicios tomar ventaja de las técnicas proporcionadas en este documento. Por lo tanto, las implementaciones del presente documento se

contemplan desde el punto de vista de una API (o de otro objeto de software), así como de un objeto de software o hardware que implementa una o más implementaciones tal como se describe en el presente documento. Por lo tanto, diversas implementaciones descritas en este documento pueden tener aspectos que son totalmente en hardware, parcialmente en hardware y parcialmente en software, así como totalmente en software.

[00113] La palabra "ejemplo" se utiliza en este documento para indicar que sirve como ejemplo, caso o ilustración. Para evitar la duda, la materia dada a conocer en el presente documento no está limitada por tales ejemplos. Además, cualquier aspecto o diseño descrito en el presente documento como "ejemplo" no debe considerarse necesariamente como preferida o ventajosa sobre otros aspectos o diseños, ni tiene la intención de impedir estructuras de ejemplo equivalentes y técnicas conocidas por los expertos en la técnica. Además, con respecto a los términos "incluye", "tiene", "contiene", y otras palabras similares que se utilizan, para evitar dudas, dichos términos están destinados a ser incluidos en una manera similar a la expresión "que comprende" como una palabra de transición abierta sin excluir ningún elemento adicional o de otros elementos cuando se emplea en una reivindicación.

[00114] Como se ha mencionado, las diversas técnicas descritas en el presente documento pueden implementarse en conexión con hardware o software o, en su caso, con una combinación de ambos. Tal como se usa en el presente documento, los términos "componente", "módulo", "sistema" y similares se pretenden igualmente para referirse a una entidad relacionada con la informática, bien sea hardware, una combinación de hardware y software, software, o software en ejecución. Por ejemplo, un componente puede ser, pero no se limita a ser, un proceso que se ejecuta en un procesador, un procesador, un objeto, un ejecutable, un hilo de ejecución, un programa, y/o un ordenador. A modo de ilustración, tanto una aplicación que se ejecuta en un ordenador y el ordenador pueden ser un

componente. Uno o más componentes pueden residir dentro de un proceso y/o hilo de ejecución y un componente puede estar localizado en un ordenador y/o distribuido entre dos o más ordenadores.

[00115] Los sistemas mencionados anteriormente se han descrito con respecto a la interacción entre varios componentes. Se puede apreciar que tales sistemas y componentes pueden incluir aquellos componentes o subcomponentes especificados, algunos de los componentes o subcomponentes especificados, y/o componentes adicionales, y de acuerdo con diversas permutaciones y combinaciones de los anteriores. Los subcomponentes también pueden ser implementados como componentes acoplados comunicativamente a otros componentes en lugar de componentes incluidos en los padres (jerárquicos). Además, puede observarse que uno o más componentes pueden combinarse en un único componente que proporciona la funcionalidad agregada o dividida en varios subcomponentes separados, y que cualquier una o más capas intermedias, tales como una capa de gestión, se pueden proporcionar para acoplar comunicativamente tales subcomponentes con el fin de proporcionar una funcionalidad integrada. Cualquiera de los componentes descritos en este documento también puede interactuar con uno o más de otros componentes no descritos específicamente en este documento pero conocidos en general por los expertos en la técnica.

[00116] En vista de los sistemas de ejemplo descritos en el presente documento, las metodologías que pueden ser implementadas de acuerdo con la materia objeto descrita también pueden ser apreciados con referencia a las gráficas de flujo/diagramas de flujo de las diversas figuras. Mientras que por motivos de simplicidad de la explicación, las metodologías se muestran y describen como una serie de bloques, es de entenderse y apreciarse que las diversas implementaciones no están limitadas por el orden de los bloques, ya que algunos bloques pueden ocurrir en diferentes órdenes y/o concurrentemente con otros bloques de lo que se

muestra y describe en este documento. Donde el diagrama de flujo no es secuencial o ramificado, el flujo se ilustra mediante gráficas de flujo/diagramas de flujo, se puede apreciar que varias otras ramas, trayectorias de flujo, y los órdenes de los bloques, pueden ser implementados para lograr el mismo resultado o uno similar.

5 Además, algunos bloques ilustrados son opcionales en la aplicación de las metodologías descritas en el presente documento.

CONCLUSIÓN

[00117] Aunque la invención es susceptible de diversas modificaciones y
10 construcciones alternativas, ciertas implementaciones ilustradas de la misma se muestran en los dibujos y se han descrito anteriormente en detalle. Debe entenderse, sin embargo, que no hay intención de limitar la invención a las formas específicas descritas, sino por el contrario, la intención es cubrir todas las modificaciones, construcciones alternativas y equivalentes que caen dentro del
15 espíritu y alcance de la invención.

[00118] Además de las diversas implementaciones descritas en este documento, es de entenderse que otras implementaciones similares pueden ser utilizadas o modificaciones y adiciones se pueden hacer a la aplicación (es) descrita para realizar la misma función o una equivalente de la aplicación correspondiente (s) sin
20 desviarse de la misma. Aún más, múltiples chips de procesamiento o varios dispositivos pueden compartir el rendimiento de una o más funciones descritas en este documento, y de manera similar, el almacenamiento puede efectuarse a través de una pluralidad de dispositivos. Por consiguiente, la invención no ha de limitarse a una sola aplicación, sino más bien debe interpretarse en amplitud, espíritu y
25 alcance de acuerdo con las reivindicaciones adjuntas.