

## **TECNOLOGÍAS PARA ALMACENAMIENTO EN CACHÉ CONFIGURABLE,**

### **DEDUPLICACIÓN Y MANEJO DE LÍMITES DE VELOCIDAD**

#### **REIVINDICACIÓN DE PRIORIDAD**

Esta solicitud reivindica el beneficio de la solicitud de patente de los EE. UU. núm. 18/345.243, presentada el 30 de junio de 2023, titulada “TECHNOLOGIES FOR CONFIGURABLE CACHING, DEDUPLICATION, AND RATE LIMIT HANDLING”.

#### **ANTECEDENTES**

El desarrollo web tradicionalmente sigue una fórmula de creación de aplicaciones de una sola página (SPA, por sus siglas en inglés) o aplicaciones de múltiples páginas en las que cada página implementa su propio manejo para solicitudes de interfaz de programación de aplicaciones (API, por sus siglas en inglés), almacenamiento en caché, configuraciones, permisos, eventos, y registro. Los almacenamientos de datos a nivel de página son soluciones comunes, pero están vinculados a entornos específicos y requieren revisión cuando se implementan soluciones en múltiples entornos. Este enfoque provoca la duplicación de código en múltiples repositorios pequeños, o todo el código dependiendo del manejo que se vaya a realizar en

el mismo repositorio. La duplicación de código provoca la duplicación de trabajo durante la depuración, las actualizaciones de dependencia, o el mantenimiento regular, así como inconsistencias entre las páginas. Por otro lado, tener todo el código que dependa de la capa de servicio dentro de un repositorio provoca repositorios grandes, complejos, y difíciles de administrar que ralentizan el proceso de mantenimiento de características existentes y el desarrollo de nuevas características. Los componentes también deben reconstruirse cuando se mueven entre páginas y entornos de interfaz de usuario (p. ej., debido a diferencias de implementación de lógica a nivel de página o entorno).

## **SUMARIO**

Una modalidad se dirige a un sistema, componentes, y métodos únicos para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de las solicitudes de datos de interfaz de usuario. Otras modalidades se dirigen a aparatos, sistemas, dispositivos, hardware, métodos, y combinaciones de estos para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario.

De acuerdo con una modalidad, un método para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario puede incluir recibir una solicitud de datos iniciada por un elemento de interfaz de usuario de una interfaz de usuario, determinar si una caché de deduplicación incluye una entrada de deduplicación asociada con la solicitud de datos, determinar si una caché principal, diferente de la caché de deduplicación, incluye una entrada de caché asociada con la solicitud de datos en respuesta a determinar que la caché de deduplicación no incluye la entrada de duplicación, determinar si la solicitud de datos está asociada con un grupo limitado por velocidad de puntos finales de interfaz de programación de aplicaciones en respuesta a uno de (i) determinar que la caché principal no incluye la entrada de caché asociada con la solicitud de datos o (ii) determinar que la entrada de caché asociada con la solicitud de datos ha caducado, y realizar la solicitud de datos a un sistema en modo administrador después de que haya transcurrido un período de reintento de servidor predefinido asociado con el grupo limitado por velocidad en respuesta a determinar que la solicitud de datos está asociada con el grupo limitado por velocidad.

En algunas modalidades, la entrada de deduplicación puede indicar que una solicitud de datos

anterior idéntica a la solicitud de datos iniciada por el elemento de interfaz de usuario de la interfaz de usuario se realizó dentro de un período de tiempo umbral.

En algunas modalidades, el método puede incluir además determinar si la solicitud de datos está asociada con una antigüedad de la caché máxima predefinida en respuesta a determinar que la caché principal incluye la entrada de caché asociada con la solicitud de datos.

En algunas modalidades, el método puede incluir además determinar si la entrada de caché asociada con la solicitud de datos ha caducado basado en un tiempo de creación de la entrada de caché y la antigüedad de la caché máxima predefinida en respuesta a determinar que la solicitud de datos está asociada con la antigüedad de la caché máxima predefinida.

En algunas modalidades, el método puede incluir además determinar si la entrada de caché asociada con la solicitud de datos ha caducado basado en un tiempo de caducidad de la entrada de caché definido por la entrada de caché en respuesta a determinar que la solicitud de datos no está asociada con la antigüedad de la caché máxima predefinida.

En algunas modalidades, el método puede incluir realizar la solicitud de datos al sistema en modo administrador

inmediatamente en respuesta a determinar que la solicitud de datos no está asociada con ningún grupo limitado por velocidad.

En algunas modalidades, el método puede incluir además recibir un error asociado con la solicitud de datos en respuesta a un incumplimiento de la solicitud de datos, determinar si el error asociado con la solicitud de datos es un error recuperable, determinar si se ha alcanzado un límite de reintentos predefinido en respuesta a determinar que el error asociado con la solicitud de datos es el error recuperable, y reintentar la solicitud de datos en respuesta a determinar que no se ha alcanzado el límite de reintentos predefinido.

En algunas modalidades, reintentar la solicitud de datos puede incluir reintentar la solicitud de datos después del período de reintento de servidor predefinido en respuesta a determinar que el error recuperable es un error 429.

En algunas modalidades, el límite de reintentos predefinido puede ser un número máximo de reintentos que se pueden realizar.

En algunas modalidades, determinar si la solicitud de datos está asociada con el grupo limitado por velocidad de puntos finales de interfaz de programación de aplicaciones puede incluir determinar si un punto final de interfaz de programación de aplicaciones al que se dirige la solicitud de

datos se incluye en el grupo limitado por velocidad de puntos finales de interfaz de programación de aplicaciones.

En algunas modalidades, la caché de deduplicación puede configurarse para almacenar una o más promesas, en donde cada una de las promesas o más está asociada con una solicitud de datos respectiva anterior.

En algunas modalidades, la caché de deduplicación puede configurarse para almacenar datos devueltos en respuesta a solicitudes de datos anteriores con un tiempo de espera configurable.

En algunas modalidades, la caché principal puede configurarse para almacenar datos devueltos en respuesta a solicitudes de datos anteriores.

De acuerdo con otra modalidad, un sistema para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario puede incluir una caché de deduplicación, una caché principal diferente de la caché de deduplicación, al menos un procesador, y al menos una memoria que comprende una pluralidad de instrucciones almacenadas en la misma que, en respuesta a la ejecución por al menos un procesador, hace que el sistema reciba una solicitud de datos iniciada por un elemento de interfaz de usuario de una interfaz de usuario, determine si la caché de deduplicación incluye una entrada de

deduplicación asociada con la solicitud de datos, determine si la caché principal incluye una entrada de caché asociada con la solicitud de datos en respuesta a una determinación de que la caché de deduplicación no incluye la entrada de deduplicación, determine si la solicitud de datos está asociada con un grupo limitado por velocidad de puntos finales de interfaz de programación de aplicaciones en respuesta a una determinación de que uno de (i) la caché principal no incluye la entrada de caché asociada con la solicitud de datos o (ii) la entrada de caché asociada con la solicitud de datos ha caducado, y realice la solicitud de datos a un sistema en modo administrador después de que haya transcurrido un período de reintento de servidor predefinido asociado con el grupo limitado por velocidad en respuesta a una determinación de que la solicitud de datos está asociada con el grupo limitado por velocidad.

En algunas modalidades, la entrada de deduplicación puede indicar que una solicitud de datos anterior idéntica a la solicitud de datos iniciada por el elemento de interfaz de usuario de la interfaz de usuario se realizó dentro de un período de tiempo umbral.

En algunas modalidades, la pluralidad de instrucciones puede hacer además que el sistema determine si la solicitud de datos está asociada con una antigüedad de la caché

máxima predefinida en respuesta a una determinación de que la caché principal incluye la entrada de caché asociada con la solicitud de datos, determine si la entrada de caché asociada con la solicitud de datos ha caducado basado en un tiempo de creación de la entrada de caché y la antigüedad de la caché máxima predefinida en respuesta a una determinación de que la solicitud de datos está asociada con la antigüedad de la caché máxima predefinida, y determine si la entrada de caché asociada con la solicitud de datos ha caducado basado en el tiempo de caducidad de la entrada de caché definido por la entrada de caché en respuesta a una determinación de que la solicitud de datos no está asociada con la antigüedad de la caché máxima predefinida.

En algunas modalidades, la pluralidad de instrucciones puede hacer además que el sistema reciba un error asociado con la solicitud de datos en respuesta a un incumplimiento con la solicitud de datos, determinar si el error asociado con la solicitud de datos es un error recuperable, determinar si se ha alcanzado un límite de reintentos predefinido en respuesta a una determinación de que el error asociado con la solicitud de datos es el error recuperable, y reintentar la solicitud de datos en respuesta a una determinación de que no se ha alcanzado el límite de reintentos predefinido.

En algunas modalidades, reintentar la solicitud de datos puede incluir reintentar la solicitud de datos después del período de reintento de servidor predefinido en respuesta a una determinación de que el error recuperable es un error 429.

En algunas modalidades, el límite de reintentos predefinido puede ser un número máximo de reintentos que se pueden realizar.

En algunas modalidades, la caché de deduplicación puede configurarse para almacenar una o más promesas, en donde cada una de la uno o más promesas está asociada con una solicitud de datos respectiva previa, y la caché principal puede configurarse para almacenar datos devueltos en respuesta a solicitudes de datos anteriores.

Este sumario no pretende identificar características clave o esenciales de la materia reivindicada, ni pretende usarse como una ayuda para limitar el alcance de la materia reivindicada. Modalidades adicionales, formas, características, y aspectos de la presente solicitud serán evidentes a partir de la descripción y las figuras proporcionadas con la presente.

#### **BREVE DESCRIPCIÓN DE LAS FIGURAS**

Los conceptos descritos en la presente descripción son ilustrativos a manera de ejemplo y no a manera de limitación en las figuras adjuntas. Para simplicidad y claridad de la ilustración, los elementos ilustrados en las figuras no están necesariamente dibujados a escala. Cuando se considere adecuado, las etiquetas actuales se repiten entre las figuras para indicar elementos correspondientes o análogos.

La Figura 1 representa un diagrama en bloque simplificado de al menos una modalidad de un sistema para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario;

la Figura 2 es un diagrama en bloque simplificado de al menos una modalidad de un dispositivo informático; y

las figuras 3-4 son un diagrama de flujo simplificado de al menos una modalidad de un método de almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario.

## **DESCRIPCIÓN DETALLADA**

Aunque los conceptos de la presente descripción son susceptibles a diversas modificaciones y formas alternativas, las modalidades específicas se han mostrado a manera de ejemplo en las figuras y se describirán en la presente descripción en detalle. Sin embargo, debe entenderse que no existe intención de limitar los conceptos de la presente descripción a las formas particulares descritas, sino que, por el contrario, la intención es cubrir todas las modificaciones, equivalentes, y alternativas consistentes con la presente descripción y las reivindicaciones adjuntas.

Las referencias en la especificación a “una modalidad”, “una modalidad ilustrativa”, etc., indican que la modalidad descrita puede incluir una característica, estructura, o funcionalidad particular, pero cada modalidad puede o no necesariamente incluir esa característica, estructura, o funcionalidad particular. Además, tales frases no necesariamente se refieren a la misma modalidad. También debe apreciarse que, aunque la referencia a un componente o característica “preferida” puede indicar la deseabilidad de un componente o característica particular con respecto a una modalidad, la descripción no es tan limitante con respecto a otras modalidades, que puede omitir tal componente o

característica. Además, cuando una característica, estructura, o funcionalidad particular se describe en conexión con una modalidad, se afirma que está dentro del conocimiento de un experto en la técnica para implementar tal característica, estructura, o funcionalidad en conexión con otras modalidades, ya sea que se describan o no explícitamente. Además, características, estructuras, o funcionalidades particulares pueden combinarse en cualquier combinación y/o subcombinaciones adecuadas en diversas modalidades.

Adicionalmente, debe apreciarse que los artículos incluidos en una lista en forma de “al menos uno de A, B, y C” puede significar (A); (B); (C); (A y B); (B y C); (A y C); o (A, B, y C). Similarmente, los artículos enumerados en la forma de “al menos uno de A, B, o C” puede significar (A); (B); (C); (A y B); (B y C); (A y C); o (A, B, y C). Además, con respecto a las reivindicaciones, el uso de palabras y frases tales como “un”, “una”, “al menos uno” y/o “al menos una porción” no deben interpretarse como que limita solo uno de tales elementos a menos que se indique específicamente lo contrario, y el uso de frases tales como “al menos una porción” y/o “una porción” deben interpretarse como que abarca ambas modalidades que incluyen solo una porción de tal elemento y modalidades que incluyen la

totalidad de tal elemento a menos que se indique específicamente lo contrario.

En algunos casos, las modalidades descritas pueden implementarse en hardware, firmware, software, o una combinación de estos. Las modalidades descritas también pueden implementarse como instrucciones transportadas o almacenadas en uno o más medios de almacenamiento transitorios o no transitorios legibles por máquina (p. ej., legibles por computador), que pueden leerse y ejecutarse mediante uno o más procesadores. Un medio de almacenamiento legible por máquina puede representarse como cualquier dispositivo de almacenamiento, mecanismo, u otra estructura física para almacenar o transmitir información en una forma legible por una máquina (p. ej., una memoria volátil o no volátil, un disco multimedia, u otro dispositivo multimedia).

En las figuras, algunas características estructurales o de método pueden mostrarse en disposiciones y/u ordenamientos específicos. Sin embargo, debe apreciarse que tales disposiciones y/u ordenamientos específicos pueden no ser necesarios. Más bien, en algunas modalidades, tales características pueden disponerse de una manera y/u orden diferente del que se muestra en las figuras ilustrativas a menos que se indique lo contrario. Adicionalmente, la inclusión de una característica estructural o de método en

una figura particular no pretende implicar que se requiere tal característica en todas las modalidades y, en algunas modalidades, puede no incluirse o puede combinarse con otras características.

Los paradigmas de desarrollo web tradicionales provocan la duplicación de trabajo, ineficiencias, e inconsistencias. Cada una de las bibliotecas existentes para la implementación del manejo de API, almacenamiento en caché, registro, y otras características generalmente se enfocan de manera limitada a una función específica y no proporcionan características clave que minimizarían las solicitudes al servicio en la nube y manejarían errores 429 dispersos del microservicio. Las soluciones de almacenamiento en caché son de muy alto nivel, genéricas, no se pueden almacenar en caché basado en los parámetros y no se pueden configurar dinámicamente. Las tecnologías existentes dan como resultado un significativo desperdicio de recursos computacionales y de desarrollo (p. ej., ciclos de CPU, RAM, almacenamiento, etc.). Tanto en la interfaz de usuario (p. ej., un dispositivo el cliente) como en el modo administrador (p. ej., un dispositivo en la nube), los recursos se desperdician por el manejo ineficiente de reintentos y solicitudes. Aunque los microservicios permiten economías de escala, ya que permiten que múltiples desarrolladores construyan e implementen una capa

de servicio, eso no permite usar una biblioteca común y volver a implementar y copiar el código. Beneficios y problemas similares están presentes en las microaplicaciones.

Debe apreciarse que las tecnologías descritas en la presente descripción permiten la orquestación de servicios comunes mediante servicios de solicitud, datos, y configuración para impulsar el desarrollo de componentes independientes. Por ejemplo, se proporcionan métodos de entrada para llamadas de API RESTful estándar, e integrados con estos puntos de entrada se pueden parametrizar el almacenamiento en caché configurable, la deduplicación, y la lógica para manejar errores o códigos de estado HTTP de demasiadas solicitudes 429. El almacenamiento en caché combinado con la deduplicación de solicitudes como se describe en la presente descripción puede proporcionar una minimización del flujo de datos para las llamadas de API RESTful, lo que mejora el rendimiento de la interfaz de usuario (UI, por sus siglas en inglés) y permite que múltiples componentes de la interfaz de usuario se implementen individualmente sin preocupaciones con respecto a la administración del flujo de API a la capa de servicio en la nube. Debe apreciarse además que las tecnologías descritas en la presente descripción pueden usarse como una capa de servicio común, lo que permite que los componentes

de la interfaz de usuario se muevan libremente entre microaplicaciones debido a la naturaleza autónoma y agnóstica del entorno web de la solución. Adicionalmente, las tecnologías permiten reducir volver a solicitar datos de la capa de servicio, disminuyendo el procesamiento a través de la pila de la nube y haciendo que toda la solución sea más eficiente y sostenible.

Además, como se describe en la presente descripción, los envoltorios de API modulares y autónomos pueden usarse para encapsular la lógica establecida sobre las API básicas para manejar interacciones de API más complejas y manipulación de datos. En particular, los métodos RESTful pueden manejarse de una manera común (p. ej., codificados una vez) en un objeto base donde el almacenamiento en caché y los reintentos como se describen en la presente descripción pueden aprovecharse en estas llamadas. Esto garantiza que los implementadores de las API particulares no tengan que preocuparse o volver a implementar el reintento, la deduplicación, o el manejo básico de los métodos RESTful. Por ejemplo, una API particular puede implicar una operación POST que después devuelve de forma asíncrona una URL de archivo inactivo. En algunas modalidades, todo el flujo de llamadas puede estar encapsulado en un envoltorio de API, una vez, de tal manera que los desarrolladores no tengan que volver a implementar ese

comportamiento. En otra modalidad ilustrativa, una API particular puede devolver los identificadores de usuario, pero el desarrollador final desea el nombre del usuario. En tales modalidades, las llamadas de API adecuadas pueden emparejarse con el almacenamiento en caché para exponer respuestas de nombre de usuario, en lugar de identificadores de usuario, lo que garantiza una codificación común, consistente, y simplificada desde la perspectiva del desarrollador de la interfaz de usuario. Debe apreciarse además que los envoltorios de API permiten que las API se agrupen y se escriban en archivos individuales de tal manera que solo las API que necesita una interfaz de usuario se incorporen en el paquete, lo que garantiza que los tamaños se minimicen. Adicionalmente, los desarrolladores conocedores de API específicas pueden escribir los controladores una vez, abordando cualquier comportamiento especial de almacenamiento en caché o limitación de velocidad que se deba considerar en el momento del desarrollo, y reduciendo la duplicación de código y las fallas.

Como se describe a continuación, por ejemplo en el método 300 de las Figuras 3-4, el almacenamiento en caché configurable, la deduplicación, y/o la lógica 429 pueden usarse conjuntamente entre sí. Por ejemplo, el almacenamiento en caché configurable parametrizado puede aprovecharse porque las respuestas de datos de API RESTful

dependen en gran medida del contexto, la consulta, y/o el cuerpo de la solicitud. En consecuencia, el almacenamiento en caché configurable aprovecha diversos parámetros de entrada junto con controles sobre la vida útil (p. ej., tiempo de vida (TTL, por sus siglas en inglés)) de las entradas de caché para permitir configuraciones globales, agrupadas, o individuales para las solicitudes. En otras palabras, en algunas modalidades, el almacenamiento en caché puede configurarse para cada aplicación, para cada punto final, y/o para cada tipo individual de solicitud. Adicionalmente, el servicio puede tener una configuración predeterminada para el TTL establecido en el nivel de página, que puede ser anulada de forma individual.

En algunas modalidades, las agrupaciones de caché pueden usarse para agrupar conjuntos de datos, lo que permite realizar operaciones de conjunto en la caché tal como la invalidación manual de datos vinculados. Debe apreciarse que tales características pueden ser útiles en la administración de conjuntos de solicitudes interrelacionadas, tal como un grupo de solicitudes que recuperan información de o sobre una única entidad. Por ejemplo, las agrupaciones de caché pueden permitir que la etapa posterior de una solicitud que se sabe que afecta a las entradas de caché de las solicitudes cambie esas entradas de caché como una única operación en lugar de operaciones de

entrada de caché individuales. En consecuencia, las agrupaciones de caché pueden permitir operaciones de caché que sean más rápidas de escribir y/o de cualquiera otra manera sean más eficientes. Adicionalmente, las agrupaciones de límite de velocidad y los parámetros de antigüedad de la caché máxima también se pueden usar como se describe con mayor detalle a continuación.

La lógica de reintento con lógica avanzada 429 puede implementarse para manejar los límites de velocidad variados y a veces interconectados de una arquitectura de microservicios. Por ejemplo, la agrupación de límites de velocidad se puede usar junto con la lógica avanzada 429 para “reenviar” errores 429 a todas las solicitudes afectadas, lo que impide que se realice una solicitud dentro de una agrupación que ha recibido un código de estado 429 antes de que haya transcurrido el tiempo de espera asociado. Debe apreciarse que cada punto final de API puede definir un límite de velocidad separado, y los límites de velocidad y las agrupaciones de límites de velocidad pueden depender del servicio de modo administrador. Los puntos finales de API particulares asociados con o asignados a una agrupación de límites de velocidad particular pueden ser predefinidos por un experto en el dominio, por ejemplo, a medida que se desarrolla un envoltorio de API. De esta manera, la

velocidad y la eficiencia de la interfaz de usuario pueden mejorarse al evitar que se realicen solicitudes vinculadas a fallos. Adicionalmente, como se describe con mayor detalle a continuación, la lógica 429 puede hacer que las solicitudes bloqueadas por un código de estado 429 se consoliden, reduciendo las instancias de errores 429 duplicados de las solicitudes en espera.

Debe apreciarse que las diferentes secciones de API pueden tener una lógica de reintento diferente. Por ejemplo, algunas API podrían considerar que diferentes códigos de error son recuperables solo debido a la implementación específica. Las agrupaciones de reintentos pueden aprovecharse para agrupar solicitudes de API que comparten un comportamiento de reintento común. Al hacerlo, se puede proporcionar una lógica de reintento extensible que se puede personalizar para adaptarse a las necesidades de un grupo de puntos finales. Por ejemplo, un grupo de solicitudes con un error compartido puede solucionarse con una solicitud adicional realizada como parte del proceso de reintento. En diversas modalidades, para cada una de las agrupaciones de reintentos, se puede definir lo que constituye un error recuperable, junto con las acciones particulares que se realizarán para reintentar y errores y/u otras características.

Como se describe con mayor detalle a continuación, el sistema puede aprovechar técnicas de deduplicación para consolidar solicitudes idénticas en una única solicitud dentro de un período configurable de tal manera que las solicitudes solo se manejen una sola vez. En otras palabras, las técnicas de duplicación funcionan efectivamente como una caché de deduplicación de corta duración. Aunque no es necesariamente una “caché” per se, la caché de deduplicación se describe en la presente descripción como tal para fines de simplicidad y brevedad de la descripción. En la modalidad ilustrativa, la caché de deduplicación almacena promesas asociadas con solicitudes de datos en lugar de los objetos de datos reales solicitados (que pueden almacenarse en una “caché principal” u otra memoria). La deduplicación garantiza que si los elementos de la interfaz de usuario no se posponen, las solicitudes no envían spam al servidor en modo administrador. Esto permite que diferentes componentes de la interfaz de usuario se desarrollen independientemente sin conocimiento unos de otros y sin preocuparse por realizar múltiples llamadas idénticas al servidor en modo administrador.

En algunas modalidades, las estructuras de configuración de página se pueden establecer a nivel de página, pero también se pueden anular si una aplicación específica requiere opciones personalizadas. Dependiendo de la aplicación

particular, una microaplicación puede utilizar diferentes patrones de comportamiento. Por ejemplo, puede ser deseable que los datos se almacenen en caché durante solo 10 segundos en lugar de 120 segundos. En consecuencia, la página puede ser capaz de configurar los patrones de comportamiento en un nivel superior, e impacta a todos los componentes individualmente. En consecuencia, incluso si se desarrollan una docena de componentes, no es necesario conocer la ventana de caché, ya que la página conoce la ventana de caché y la atribuye a los componentes individuales (p. ej., empujando hacia abajo configuraciones de alto nivel si no se anulan en el nivel inferior). Como tal, debe apreciarse que el esquema de configuración de página pueda proporcionar una configuración predeterminada para un desarrollo rápido, mientras que también permite la configuración individualizada de funciones de conmutación, servicio de permisos, servicio de datos, servicio de notificación, reintentos, limpieza de deduplicación, caché a nivel de aplicación, y/u otras características o parámetros.

Haciendo referencia ahora a la Figura 1, un sistema 100 para almacenamiento en caché configurable, deduplicación, y manejo de límites de velocidad de solicitudes de datos de interfaz de usuario puede incluir un sistema de interfaz de usuario 102, una red 104 y un sistema en modo administrador 106. Adicionalmente, en la modalidad ilustrativa, el sistema

de interfaz de usuario 102 incluye una caché principal 108 y una caché de deduplicación 110, y el sistema en modo administrador 106 incluye interfaces de programación de aplicaciones (API) 112.

Aunque solo un sistema de interfaz de usuario 102, una red 104 y un sistema en modo administrador 106 se muestran en la modalidad ilustrativa de la Figura 1, el sistema 100 puede incluir múltiples sistemas de interfaz de usuario 102, redes 104 y/o sistemas en modo administrador 106 en otras modalidades. Por ejemplo, en algunas modalidades, pueden usarse múltiples sistemas en modo administrador 106 para realizar las diversas funciones descritas en la presente descripción. Además, en algunas modalidades, uno o más de los sistemas descritos en la presente descripción pueden excluirse del sistema 100, uno o más de los sistemas descritos como independientes pueden formar una porción de otro sistema, y/o uno o más del sistema descrito como que forma una porción de otro sistema puede ser independiente. Adicionalmente, debe apreciarse que la arquitectura del sistema representado por el sistema 100 en la Figura 1 es solo para fines ilustrativos, y el sistema 100 puede incluir dispositivos/sistemas adicionales y/o alternativos en otras arquitecturas del sistema. Además, debe apreciarse que cada uno de los sistemas de interfaz de

usuario 102, la red 104 y/o el sistema en modo administrador 106 puedan incluir múltiples dispositivos y/o subsistemas. Por ejemplo, en algunas arquitecturas de sistema, el sistema en modo administrador 106 puede incluir al menos un servidor de aplicaciones y un servidor de base de datos. En modalidades basadas en la nube, el sistema de interfaz de usuario 102 y/o el sistema en modo administrador 106, o porciones de este, pueden incorporarse como una solución informática ambigua de servidor similar a la que se describe a continuación.

El sistema de interfaz de usuario 102 se puede incorporar como uno o más tipos de dispositivos/sistemas capaces de realizar las funciones descritas en la presente descripción. Por ejemplo, en la modalidad ilustrativa, el sistema de interfaz de usuario 102 incluye un dispositivo cliente configurado para lanzar una interfaz de usuario con la que un usuario puede interactuar. A medida que el usuario interactúa con la interfaz de usuario, un elemento de la interfaz de usuario puede iniciar una solicitud de datos dirigida a una API particular 112 del sistema en modo administrador 106 (p. ej., en respuesta a que el usuario haga clic en un elemento de interfaz de usuario tal como un botón). En la modalidad ilustrativa, el sistema de interfaz

de usuario 102 incluye una caché principal 108 y una caché de deduplicación 110.

La caché principal 108 está configurada para almacenar resultados de datos de solicitudes de datos realizadas a través de la interfaz de usuario (p. ej., objetos de datos), mientras que la caché de deduplicación 110 almacena promesas, lo que esencialmente solo indica que tendrá un resultado en algún momento pero sin referencias a la caché principal 108. En algunas modalidades, como se describe en la presente descripción, la caché de deduplicación 110 almacena datos devueltos en respuesta a solicitudes de datos anteriores con un tiempo de espera configurable. Debe apreciarse que una promesa es un objeto que representa la finalización o fallo eventual de una operación asíncrona y su valor resultante más que el valor en sí. Como se describe a continuación, la caché de deduplicación 110 puede aprovecharse para garantizar que si se realizan múltiples solicitudes de datos idénticos dentro de un corto período de tiempo predefinido, esas solicitudes de datos se consoliden en una única solicitud para evitar enviar spam al sistema en modo administrador 106. Debe apreciarse que cada uno del caché principal 108 y caché de deduplicación 110 pueden incorporarse como, o incluir, cualquier tipo de memoria, almacenamiento de datos, y/o

estructura de datos para almacenar datos y realizar las funciones descritas en la presente descripción. Además, aunque la caché principal 108 y la caché de deduplicación 110 se representan formando una porción del sistema de interfaz de usuario 102, debe apreciarse que la caché principal 108 y/o la caché de deduplicación 110 pueden formar una porción de uno o más dispositivos/sistemas del sistema 100 en otras modalidades.

La red 104 puede representarse como uno o más tipos de redes de comunicación que son capaces de facilitar la comunicación entre los diversos dispositivos conectados comunicativamente a través de la red 104. Como tal, la red 104 puede incluir una o más redes, enrutadores, interruptores, puntos de acceso, concentradores, computadores, y/u otros dispositivos de red intervinientes. Por ejemplo, la red 104 puede representarse como o incluir de cualquier otra manera, una o más redes celulares, redes telefónicas, redes de área local o amplia, redes globales disponibles públicamente (p. ej., la Internet), redes ad hoc, enlaces de comunicaciones de intervalo corto, o una combinación de estos. En algunas modalidades, la red 104 puede incluir una red de datos o voz con conmutación de circuitos, una red de datos o voz con conmutación de paquetes, y/o cualquier otra red capaz de transportar voz

y/o datos. En particular, en algunas modalidades, la red 104 puede incluir redes basadas en protocolo de Internet (IP, por sus siglas en inglés) y/o modo de transferencia asíncrono (ATM, por sus siglas en inglés). En algunas modalidades, la red 104 puede atender tráfico de voz (p. ej., a través de una red de voz sobre IP (VOIP, por sus siglas en inglés)), tráfico web (p. ej., tal como el tráfico del protocolo de transferencia de hipertexto (HTTP, por sus siglas en inglés) y el tráfico del lenguaje de marcado de hipertexto (HTML, por sus siglas en inglés)), y/u otro tráfico de red dependiendo de la modalidad particular y/o dispositivos del sistema 100 en comunicación entre sí. En diversas modalidades, la red 104 puede incluir redes cableadas e inalámbricas analógicas o digitales (p. ej., redes IEEE 802.11, red telefónica pública conmutada (PSTN, por sus siglas en inglés), red digital de servicios integrados (ISDN), y línea de suscriptor digital (xDSL, por sus siglas en inglés)), redes de telecomunicaciones móviles de tercera generación (3G, por sus siglas en inglés), redes de telecomunicaciones móviles de cuarta generación (4G, por sus siglas en inglés), redes de telecomunicaciones móviles de quinta generación (5G, por sus siglas en inglés), una red Ethernet cableada, una red privada (p. ej., una intranet), radio, televisión, cable, satélite, y /o cualquier otro

mecanismo de entrega o entunelado para transportar datos, o cualquier combinación adecuada de tales redes. La red 104 puede permitir las conexiones entre los diversos dispositivos/sistemas 102, 106 del sistema 100. Debe apreciarse que los diversos dispositivos/sistemas 102, 106 pueden comunicarse entre sí a través de diferentes redes 104 dependiendo de los dispositivos/sistemas de fuente y/o destino 102, 106.

El sistema en modo administrador 106 se puede incorporar como uno o más tipos de dispositivos/sistemas capaces de realizar las funciones descritas en la presente descripción. Como se muestra, el sistema en modo administrador 106 incluye diversas API 112 y puntos finales de API correspondientes desde los cuales el sistema de interfaz de usuario 102 realiza solicitudes de datos. Como se describió anteriormente, debe apreciarse que el número y tipo de dispositivos incluidos en el sistema en modo administrador 106 puede variar dependiendo de la arquitectura particular del sistema 100. Debe apreciarse que cada uno de los sistemas de interfaz de usuario 102, la red 104 y el sistema en modo administrador 106 pueden incorporarse como, ser ejecutados por, formar una porción de, o estar asociados con cualquier tipo de dispositivo/sistema, colección de dispositivos/sistemas, y/o

porción(es) de estos adecuada(s) para realizar las funciones descritas en la presente descripción (p. ej., el dispositivo informático 200 de la Figura 2).

Con referencia ahora a la Figura 2, se muestra un diagrama en bloque simplificado de al menos una modalidad de un dispositivo informático 200. El dispositivo informático ilustrativo 200 ilustra al menos una modalidad de cada uno de los dispositivos informáticos, sistemas, servidores, controladores, conmutadores, puertas de enlace, motores, módulos, y/o componentes informáticos descritos en la presente descripción (p. ej., a los que colectivamente puede hacerse referencia indistintamente como dispositivos informáticos, servidores, o módulos para brevedad de la descripción). Por ejemplo, los diversos dispositivos informáticos pueden ser un proceso o hilo que se ejecuta en uno o más procesadores de uno o más dispositivos informáticos 200, que pueden ejecutar instrucciones de programas informáticos e interactuar con otros módulos del sistema para realizar las diversas funcionalidades descritas en la presente descripción. A menos que se limite específicamente de cualquier otra manera, la funcionalidad descrita en relación con una pluralidad de dispositivos informáticos puede integrarse en un único dispositivo informático, o las diversas funcionalidades descritas en relación con un único dispositivo informático pueden

distribuirse a través de varios dispositivos informáticos. Además, en relación con los sistemas informáticos descritos en la presente descripción, los diversos servidores y dispositivos informáticos de estos pueden estar ubicados en dispositivos informáticos locales 200, dispositivos informáticos remotos 200 (p. ej., fuera del sitio o en un entorno informático basado en la nube o en la nube, por ejemplo, en un centro de datos remoto conectado a través de una red), o alguna combinación de estos dependiendo de la modalidad particular. En algunas modalidades, puede accederse a la funcionalidad proporcionada por servidores ubicados en dispositivos informáticos fuera del sitio y puede proporcionarse a través de una red privada virtual (VPN, por sus siglas en inglés), como si tales servidores estuvieran en el sitio, o la funcionalidad puede proporcionarse usando un software como servicio (SaaS) a los que se accede a través de Internet usando diversos protocolos, tales como el intercambio de datos a través del lenguaje de marcado extensible (XML, por sus siglas en inglés), JSON, y/o la funcionalidad puede ser accedida/aprovechada de cualquier otra manera.

En algunas modalidades, el dispositivo informático 200 puede incorporarse como un servidor, computador de escritorio, computador portátil, computador de tableta, notebook, netbook, Ultrabook™, teléfono celular, dispositivo informático móvil, teléfono inteligente, dispositivo

informático *wearable*, asistente digital personal, dispositivo de Internet de las cosas (IoT, por sus siglas en inglés), sistema de procesamiento, punto de acceso inalámbrico, enrutador, puerta de enlace, y/o cualquier otro dispositivo informático, de procesamiento, y/o de comunicación capaz de realizar las funciones descritas en la presente descripción.

El dispositivo informático 200 incluye un dispositivo de procesamiento 202 que ejecuta algoritmos y/o datos de procesos de acuerdo con la lógica operativa 208, un dispositivo de entrada/salida 204 que permite la comunicación entre el dispositivo informático 200 y uno o más dispositivos externos 210, y memoria 206 que almacena, por ejemplo, datos recibidos del dispositivo externo 210 a través del dispositivo de entrada/salida 204.

El dispositivo de entrada/salida 204 permite que el dispositivo informático 200 se comuniquen con el dispositivo externo 210. Por ejemplo, el dispositivo de entrada/salida 204 puede incluir un transceptor, un adaptador de red, una tarjeta de red, una interfaz, uno o más puertos de comunicación (p. ej., un puerto USB, puerto serie, puerto paralelo, un puerto analógico, un puerto digital, VGA, DVI, HDMI, FireWire, CAT 5, o cualquier otro tipo de puerto o interfaz de comunicación), y/u otros sistemas de circuitos de comunicación. El sistema de circuitos de comunicación del

dispositivo informático 200 puede configurarse para usar una cualquiera o más tecnologías de comunicación (p. ej., comunicaciones inalámbricas o cableadas) y protocolos asociados (p. ej., Ethernet, Bluetooth®, Wi-Fi®, WiMAX, etc.) para efectuar tal comunicación dependiendo del dispositivo informático particular 200. El dispositivo de entrada/salida 204 puede incluir hardware, software, y/o firmware adecuado para realizar las técnicas descritas en la presente descripción.

El dispositivo externo 210 puede ser cualquier tipo de dispositivo que permite que los datos sean ingresados o emitidos desde el dispositivo informático 200. Por ejemplo, en diversas modalidades, el dispositivo externo 210 puede incorporarse como uno o más de los dispositivos/sistemas descritos en la presente descripción, y/o una porción de estos. Además, en algunas modalidades, el dispositivo externo 210 puede incorporarse como otro dispositivo informático, conmutador, herramienta de diagnóstico, controlador, impresora, pantalla, alarma, dispositivo periférico (p. ej., teclado, mouse, pantalla táctil, etc.), y/o cualquier otro dispositivo informático, de procesamiento, y/o de comunicación capaz de realizar las funciones descritas en la presente descripción. Además, en algunas modalidades, debe apreciarse que el

dispositivo externo 210 puede integrarse en el dispositivo informático 200.

El dispositivo de procesamiento 202 puede incorporarse como cualquier tipo de procesador(es) capaz de realizar las funciones descritas en la presente descripción. En particular, el dispositivo de procesamiento 202 puede incorporarse como uno o más procesadores de un solo núcleo o de múltiples núcleos, microcontroladores, u otro procesador o circuitos de procesamiento/control. Por ejemplo, en algunas modalidades, el dispositivo de procesamiento 202 puede incluir o puede incorporarse como una unidad aritmética lógica (ALU, por sus siglas en inglés), unidad central de procesamiento (CPU, por sus siglas en inglés), procesador de señal digital (DSP, por sus siglas en inglés), unidad de procesamiento de gráficos (GPU, por sus siglas en inglés), matriz de puertas programables por campo (FPGA, por sus siglas en inglés), circuito integrado de aplicación específica (ASIC, por sus siglas en inglés) y/u otro(s) procesador(es) adecuado(s). El dispositivo de procesamiento 202 puede ser de un tipo programable, una máquina de estado cableada dedicada, o una combinación de estos. Los dispositivos de procesamiento 202 con múltiples unidades de procesamiento pueden utilizar procesamiento distribuido, de canalización de datos (*pipelined*), y/o paralelo en diversas modalidades. Además, el

dispositivo de procesamiento 202 puede estar dedicado a la realización de solo las operaciones descritas en la presente descripción, o puede utilizarse en una o más aplicaciones adicionales. En la modalidad ilustrativa, el dispositivo de procesamiento 202 es programable y ejecuta algoritmos y/o procesa datos de acuerdo con la lógica operativa 208 según lo definido por instrucciones de programación (tales como software o firmware) almacenadas en la memoria 206. Adicional o alternativamente, la lógica operativa 208 para el dispositivo de procesamiento 202 puede estar al menos parcialmente definida por lógica cableada u otro hardware. Además, el dispositivo de procesamiento 202 puede incluir uno o más componentes de cualquier tipo adecuado para procesar las señales recibidas desde el dispositivo de entrada/salida 204 o desde otros componentes o dispositivos y para proporcionar las señales de salida deseadas. Tales componentes pueden incluir sistemas de circuitos digitales, circuitos analógicos, o una combinación de estos.

La memoria 206 puede ser de uno o más tipos de medios no transitorios legibles por computador, tales como una memoria de estado sólido, memoria electromagnética, memoria óptica, o una combinación de estas. Además, la memoria 206 puede ser volátil y/o no volátil y, en algunas modalidades, parte de o toda la memoria 206 puede ser de un

tipo portátil, tal como un disco, cinta, memoria flash extraíble (*memory stick*), cartucho, y/u otra memoria portátil adecuada. En la operación, la memoria 206 puede almacenar diversos datos y software usados durante la operación del dispositivo informático 200, tales como sistemas operativos, aplicaciones, programas, bibliotecas, y controladores. Debe apreciarse que la memoria 206 puede almacenar datos que son manipulados por la lógica operativa 208 del dispositivo de procesamiento 202, tales como, por ejemplo, datos representativos de señales recibidas de y/o enviadas al dispositivo de entrada/salida 204 además de o en lugar de almacenar instrucciones de programación que definen la lógica operativa 208. Como se muestra en la Figura 2, la memoria 206 puede incluirse con el dispositivo de procesamiento 202 y/o acoplarse al dispositivo de procesamiento 202 dependiendo de la modalidad particular. Por ejemplo, en algunas modalidades, el dispositivo de procesamiento 202, la memoria 206, y/u otros componentes del dispositivo informático 200 pueden formar una porción de un sistema en un chip (SoC, por sus siglas en inglés) e incorporarse en un único chip de circuito integrado.

En algunas modalidades, diversos componentes del dispositivo informático 200 (p. ej., el dispositivo de procesamiento 202 y la memoria 206) pueden acoplarse

comunicativamente por medio de un subsistema de entrada/salida, que puede incorporarse como un sistema de circuitos y/o componentes para facilitar las operaciones de entrada/salida con el dispositivo de procesamiento 202, la memoria 206, y otros componentes del dispositivo informático 200. Por ejemplo, el subsistema de entrada/salida puede representarse como, o de cualquier otra manera incluir, centros de controlador de memoria, concentradores de control de entrada/salida, dispositivos de firmware, enlaces de comunicación (es decir, enlaces punto a punto, enlaces de bus, alambres, cables, guías de luz, trazas de placas de circuito impreso, etc.), y/u otros componentes y subsistemas para facilitar las operaciones de entrada/salida.

El dispositivo informático 200 puede incluir otros componentes o componentes adicionales, tales como los que se encuentran comúnmente en un dispositivo informático típico (p. ej., diversos dispositivos de entrada/salida y/u otros componentes), en otras modalidades. Debe apreciarse además que uno o más de los componentes del dispositivo informático 200 descritos en la presente descripción pueden distribuirse a través de múltiples dispositivos informáticos. En otras palabras, las técnicas descritas en la presente descripción pueden emplearse mediante un sistema informático que incluye uno o más dispositivos informáticos. Adicionalmente, aunque

solo un único dispositivo de procesamiento 202, dispositivo de E/S 204, y memoria 206 se muestran de manera ilustrativa en la Figura 2, debe apreciarse que un dispositivo informático particular 200 puede incluir múltiples dispositivos de procesamiento 202, dispositivos de E/S 204, y/o memorias 206 en otras modalidades. Además, en algunas modalidades, más de un dispositivo externo 210 puede estar en comunicación con el dispositivo informático 200.

El dispositivo informático 200 puede ser uno de una pluralidad de dispositivos conectados por una red o conectados a otros sistemas/recursos por medio de una red. La red puede representarse como uno o más tipos de redes de comunicación que son capaces de facilitar la comunicación entre los diversos dispositivos conectados comunicativamente a través de la red. Como tal, la red puede incluir una o más redes, enrutadores, interruptores, puntos de acceso, concentradores, computadores, dispositivos de cliente, puntos finales, nodos, y/u otros dispositivos de red intervinientes. Por ejemplo, la red puede representarse como o incluir de cualquier otra manera, una o más redes celulares, redes telefónicas, redes de área local o amplia, redes globales disponibles públicamente (p. ej., internet), redes ad hoc, enlaces de comunicaciones de rango corto, o una combinación de estos. En algunas modalidades, la red puede incluir una red de datos o voz con conmutación de

circuitos, una red de datos o voz con conmutación de paquetes, y/o cualquier otra red capaz de transportar voz y/o datos. En particular, en algunas modalidades, la red puede incluir redes basadas en protocolo de Internet (IP, por sus siglas en inglés) y/o modo de transferencia asíncrono (ATM, por sus siglas en inglés). En algunas modalidades, la red puede atender tráfico de voz (p. ej., a través de una red de voz sobre IP (VOIP)), tráfico web, y/u otro tráfico de red dependiendo de la modalidad particular y/o dispositivos del sistema en comunicación entre sí. En diversas modalidades, la red puede incluir redes cableadas e inalámbricas analógicas o digitales (p. ej., redes IEEE 802.11, red telefónica pública conmutada (PSTN), red digital de servicios integrados (ISDN, por sus siglas en inglés), y línea de suscriptor digital (xDSL)), redes de telecomunicaciones móviles de tercera generación (3G), redes de telecomunicaciones móviles de cuarta generación (4G), redes de telecomunicaciones móviles de quinta generación (5G), una red Ethernet cableada, una red privada (por ejemplo, una intranet), radio, televisión, cable, satélite, y /o cualquier otro mecanismo de entrega o entunelado para transportar datos, o cualquier combinación adecuada de tales redes. Debe apreciarse que los diversos dispositivos/sistemas pueden comunicarse entre sí mediante diferentes redes dependiendo de los dispositivos/sistemas de fuente y/o destino.

Debe apreciarse que el dispositivo informático 200 puede comunicarse con otros dispositivos informáticos 200 a través de cualquier tipo de puerta de enlace o protocolo de entunelado tal como la capa de cavidad segura o la seguridad de la capa de transporte. La interfaz de red puede incluir un adaptador de red integrado, tal como una tarjeta de interfaz de red, adecuado para conectar el dispositivo informático a cualquier tipo de red capaz de realizar las operaciones descritas en la presente descripción. Además, el entorno de red puede ser un entorno de red virtual donde se virtualizan los diversos componentes de red. Por ejemplo, las diversas máquinas pueden ser máquinas virtuales implementadas como un computador basado en software que se ejecuta en una máquina física. Las máquinas virtuales pueden compartir el mismo sistema operativo o, en otras modalidades, puede ejecutarse un sistema operativo diferente en cada instancia de máquina virtual. Por ejemplo, se usa un tipo de virtualización de “hipervisor” donde varias máquinas virtuales se ejecutan en la misma máquina física host, cada una de las cuales actúa como si tuviera su propio cuadro dedicado. Otros tipos de virtualización pueden emplearse en otras modalidades, tales como, por ejemplo, la red (p. ej., a través de una red de software definida) o funciones (p. ej., mediante funciones de red virtual).

Por consiguiente, uno o más de los dispositivos informáticos 200 descritos en la presente descripción pueden incorporarse como, o formar una porción de, uno o más sistemas basados en la nube. En las modalidades basadas en la nube, el sistema basado en la nube puede representarse como una solución informática ambigua de servidor, por ejemplo, que ejecuta una pluralidad de instrucciones a pedido, contiene lógica para ejecutar instrucciones solo cuando lo solicita una actividad/lanzamiento particular, y no consume recursos informáticos cuando no está en uso. Es decir, el sistema puede representarse como un entorno informático virtual que reside “en” un sistema informático (p. ej., una red distribuida de dispositivos) en el que diversas funciones virtuales (p. ej., funciones Lambda, funciones de Azure, funciones de nube de Google, y/u otras funciones virtuales adecuadas) pueden ejecutarse con las funciones del sistema descrito en la presente descripción. Por ejemplo, cuando se produce un evento (p. ej., datos se transfieren al sistema para su atención), el entorno informático virtual puede comunicarse (p. ej., a través de una solicitud a un API del entorno informático virtual), por lo cual el API puede enrutar la solicitud a la función virtual correcta (p. ej., un recurso de computación de servidor ambiguo particular) basado en un conjunto de normas. Como tal, cuando un usuario

realiza una solicitud para la transmisión de datos (p. ej., a través de una interfaz de usuario adecuada al sistema), la(s) función(es) virtual(s) adecuada(s) puede(n) ejecutarse para realizar las acciones antes de eliminar la instancia de la(s) función(es) virtual(es).

Haciendo referencia ahora a las Figuras 3-4, en uso, el sistema 100 puede ejecutar un método 300 para el almacenamiento en caché configurable, la deduplicación, y el manejo de límites de velocidad de las solicitudes de datos de interfaz de usuario. Debe apreciarse que los bloques particulares del método 300 se ilustran a manera de ejemplo, y tales bloques pueden combinarse o dividirse, añadirse o eliminarse, y/o reordenarse total o parcialmente dependiendo de la modalidad particular, a menos que se indique lo contrario. Debido a que el sistema 100 puede tener diversas arquitecturas de sistemas diferentes, los bloques del método 300 de las Figuras 3-4 se describen principalmente como ejecutados simplemente por el sistema 100 por simplicidad y brevedad de la descripción.

El método ilustrativo 300 comienza con el bloque 302 de la Figura 3 en el que un elemento de interfaz de usuario inicia una solicitud de datos. Debe apreciarse que la solicitud de datos puede hacer referencia al punto final de API/URL relevante e incluir cualquier parámetro/dato

relevante. En el bloque 304, el sistema 100 determina si la caché de deduplicación 110 incluye una entrada de deduplicación asociada con la solicitud de datos (p. ej., indicando que se realizó una solicitud de datos idéntica anterior dentro de un período de tiempo umbral configurable). Por ejemplo, como se describió anteriormente, la caché de deduplicación 110 puede configurarse para almacenar promesas asociadas con solicitudes de datos respectivas. En consecuencia, las diversas entradas de deduplicación en la caché de deduplicación 110 son indicativas de una solicitud de datos anterior que se realiza dentro de un período de tiempo umbral (es decir, el período de caducidad para la entrada de deduplicación, después del cual puede ser eliminada de la caché de deduplicación 110). Por ejemplo, en una modalidad, el período de tiempo umbral puede ser 10 ms, mientras que en otra modalidad, el umbral puede ser 100 ms. Debe apreciarse que ciertos métodos HTTP se consideran no idempotentes y generalmente no se almacenan en caché. Sin embargo, en la modalidad ilustrativa, la caché de deduplicación 110, no obstante, las almacena en caché en el sentido de que es misma la solicitud o promesa la que se almacena en caché en lugar de los datos resultantes reales. Debe apreciarse que la caché de deduplicación 110 se aprovecha para garantizar que las solicitudes de datos duplicados no se realicen dentro de un

período de tiempo predefinido (p. ej., debido a que un usuario hace clic inadvertida y rápidamente en un elemento de la interfaz de usuario de manera repetida).

Si el sistema 100 determina, en el bloque 304, que hay una entrada de deduplicación asociada en la caché de deduplicación 110, el método 300 avanza al bloque 316 en el que se realiza la solicitud de datos y los datos se devuelven al elemento de interfaz de usuario. En otras palabras, si hay una entrada de deduplicación asociada en la caché de deduplicación 110, entonces ya se ha realizado la solicitud de datos y el sistema 100 puede esperar a que se devuelvan los datos solicitados. Sin embargo, si el sistema 100 determina, en el bloque 304, que la caché de deduplicación 110 no incluye una entrada de deduplicación asociada, el método 300 avanza al bloque 306 en el que el sistema 100 determina si la caché principal 108 incluye una entrada de caché asociada con la solicitud de datos.

Si el sistema 100 determina, en el bloque 306, que la caché principal 108 incluye una entrada de caché asociada con la solicitud de datos, el método 300 avanza al bloque 308 en el que el sistema 100 determina si la solicitud de datos está asociada con una antigüedad de la caché máxima predefinida. En otras palabras, el sistema 100 determina si la solicitud de datos incluye un parámetro de antigüedad de

la caché máxima que anula cualquier tiempo de caducidad proporcionado por la misma entrada de caché. Debe apreciarse que diferentes componentes pueden tener diferentes nociones de lo que es aceptable en términos del TTL de una respuesta de datos. Por ejemplo, un componente solo puede aceptar datos que tengan un máximo de dos minutos de antigüedad, mientras que otro componente puede aceptar datos de diez minutos de antigüedad.

Si el sistema 100 determina, en el bloque 308, que la solicitud de datos está asociada con una antigüedad de la caché máxima predefinida, el método 300 avanza al bloque 310 en el que el sistema 100 calcula la caducidad de la entrada de caché usando la antigüedad de la caché máxima. Más específicamente, el sistema 100 puede determinar si la entrada de caché asociada con la solicitud de datos ha caducado basado en el tiempo de creación de la entrada de caché y la antigüedad de la caché máxima predefinida (p. ej., la caducidad es la antigüedad de la caché máxima después del tiempo de creación de la entrada de caché). Sin embargo, si el sistema 100 determina, en el bloque 308, que la solicitud de datos no está asociada con una antigüedad de la caché máxima predefinida, el método 300 avanza al bloque 312 en el que el sistema 100 usa la caducidad de la entrada de caché asociada con la solicitud de datos, de tal manera que el sistema 100 puede determinar si la

entrada de caché asociada con la solicitud de datos ha caducado basado en el tiempo de caducidad definido por la misma entrada de caché. El método 300 avanza desde los bloques 310, 312 hasta el bloque 314 en el que el sistema 100 determina si la entrada de caché ha caducado. Si no es así, el método 300 avanza al bloque 316 en el que se realiza la solicitud de datos y los datos se devuelven al elemento de interfaz de usuario.

Volviendo al bloque 306, si el sistema 100 determina que la caché principal 108 no incluye una entrada de caché asociada con la solicitud de datos, o si el sistema 100 determina, en el bloque 314, que la entrada de caché existe pero ha caducado, entonces el método 300 avanza al bloque 318 de la Figura 4 en el que el sistema 100 determina si la solicitud de datos está asociada con un grupo limitado por velocidad de puntos finales de API. Como se describió anteriormente, los grupos limitados por velocidad de los puntos finales de API pueden ser predefinidos por el sistema 100 para ayudar a evitar la transmisión de ciertas solicitudes de datos innecesarias. Por ejemplo, en una modalidad, los puntos finales GET\_USER y GET\_USERS pueden pertenecer a la misma agrupación de velocidad limitada, de tal manera que si el sistema 100 ya sabe que la solicitud GET\_USER ha sido de velocidad limitada, entonces la

solicitud GET\_USERS debería diferirse para evitar un error 429 inevitable.

Si el sistema 100 determina, en el bloque 318, que la solicitud de datos está asociada con un grupo limitado por velocidad, el método 300 avanza al bloque 320 en el que el sistema 100 retrasa el período de tiempo de reintento del servidor correspondiente antes de realizar la solicitud de datos al sistema en modo administrador 106 en el bloque 322. Si el sistema 100 determina, en el bloque 318, que la solicitud de datos no está asociada con un grupo limitado por velocidad, el método 300 avanza al bloque 322 en el que la solicitud de datos puede realizarse sin retraso.

En el bloque 324, el sistema 100 determina si se cumple la solicitud de datos. Si es así, el método 300 vuelve al bloque 316 de la Figura 3 en el que se realiza la solicitud de datos y los datos se devuelven al elemento de interfaz de usuario. Sin embargo, si las solicitudes de datos resultan fallidas, se puede dar un código de estado o error asociado con el incumplimiento de la solicitud de datos, y el método 300 avanza al bloque 326 en el que el sistema 100 determina si el error asociado con la solicitud de datos es un error recuperable. Si no es así, el método 300 avanza al bloque 328 en el que se devuelve un error de solicitud de datos al elemento de interfaz de usuario. Sin embargo, si el

sistema 100 determina, en el bloque 326, que el error es recuperable, el método 300 avanza al bloque 330 en el que el sistema 100 determina si se ha alcanzado un límite de reintentos predefinido. Debe apreciarse que el límite de reintentos predefinido puede ser configurable, y puede definir el número máximo de reintentos que se pueden realizar. En algunas modalidades, el sistema 100 puede utilizar diferentes reintentos para diferentes circunstancias (p. ej., diferentes puntos finales de API, diferentes tipos de error, etc.). Si se ha alcanzado el límite de reintentos predefinido, el método 300 avanza al bloque 328 en el que se devuelve un error de solicitud de datos al elemento de interfaz de usuario.

Sin embargo, si no se ha alcanzado el límite de reintentos predefinido, el sistema 100 puede reintentar la solicitud de datos. Más específicamente, en la modalidad ilustrativa, el método 300 avanza al bloque 332 en el que el sistema 100 determina si el error recuperable es un error 429 (p. ej., asociado con un error o código de estado HTTP de demasiadas solicitudes 429). Si es así, el método 300 avanza al bloque 320 en el que el sistema 100 se retrasa durante un período de reintento del servidor correspondiente (p. ej., proporcionado por el error 429) antes de reintentar la solicitud de datos. Sin embargo, si el error recuperable no es

un error 429, el método 300 avanza al bloque 322 en el que se puede realizar la solicitud de datos sin retraso.

Aunque los bloques 302-332 se describen de una manera relativamente en serie, debe apreciarse que diversos bloques del método 300 pueden realizarse en paralelo en algunas modalidades.