(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 689 498**

(51) Int. CI.:

H04L 29/06 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **19.12.2003 PCT/GB2003/005598**

(87) Fecha y número de publicación internacional: **08.07.2004 WO04057828**

(96) Fecha de presentación y número de la solicitud europea: **19.12.2003 E 03782665 (8)**

(97) Fecha y número de publicación de la concesión europea: **11.07.2018 EP 1576784**

(54) Título: **Método de replicación automática de objetos de datos entre un dispositivo móvil y un servidor**

(30) Prioridad:

19.12.2002 GB 0229572

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
14.11.2018

(73) Titular/es:

OPENWAVE MESSAGING DATA CENTRE LIMITED (100.0%)
Beechfield House, Winterton Way Lyme Green Business and Retail Park
Macclesfield, SK11 0JP, GB

(72) Inventor/es:

GREENWELL, THOMAS, RALPH, EDWARDS;
SPENCE, STEPHEN, TIMOTHY y
STALKER, MARK, COLIN

(74) Agente/Representante:

RIZZO , Sergio

DESCRIPCIÓN

Método de replicación automática de objetos de datos entre un dispositivo móvil y un servidor

CAMPO DE LA INVENCIÓN

[0001] La presente invención se refiere a un método de replicación automática de objetos de datos entre un dispositivo móvil y un servidor; la replicación de datos se necesita, por ejemplo, para realizar una copia de seguridad desde el dispositivo móvil hacia el servidor y para garantizar que el dispositivo móvil tiene los datos más actualizados conservados en el servidor. El dispositivo móvil y el servidor están conectados por medio de una red inalámbrica, que puede comprender una red de amplio alcance, tal como una red de telefonía celular. La red inalámbrica también puede comprender una red de corto alcance, tal como una red 802.11 o una combinación de conexiones de corto alcance, de amplio alcance y de cable.

DESCRIPCIÓN DE LA TÉCNICA ANTERIOR

[0002] En el documento de patente WO 01/78319, se expone un sistema y método para insertar artículos de datos seleccionados por el usuario desde un sistema anfitrión hacia un dispositivo de comunicación de datos móvil de un usuario tras detectar que se han producido uno o varios desencadenadores de eventos definidos por el usuario.

[0003] En el documento de patente EP 0794646, se expone un sistema de gestión de datos y un método para gestionar copias de un archivo de datos compartido mantenido en una pluralidad de sistemas informáticos que pueden conectarse a través de una red de comunicación móvil.

SUMARIO DE LA PRESENTE INVENCIÓN

[0004] En las reivindicaciones adjuntas, se establecen los aspectos de la invención.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0005] La presente invención se describirá con referencia a los dibujos adjuntos, en los que se muestran gráficos de cómo pueden variar los parámetros utilizados para controlar la replicación de datos con el tiempo.

DESCRIPCIÓN DETALLADA

[0006] La presente invención es aplicada por Cognima Ltd (Londres, Reino Unido) para posibilitar que los operadores de red móvil puedan controlar la temporización de la replicación de datos en el sistema Cognima Replicate™. El presente documento da por sentado el conocimiento del funcionamiento del sistema Cognima Replicate™, que se describe con mayor detalle en el **Apéndice 1**.

[0007] Cabe observar que el término calidad de servicio o *QoS*, por sus siglas en inglés, se utiliza a lo largo del presente documento, pero salvo que se afirme en un contexto, no está relacionado con el significado técnico de *QoS* en el contexto de RFC2475. Este documento IETF define el término *QoS* de forma muy precisa en términos de un número de métricas de la capa IP, considerando que la primera implementación de *QoS* de Cognima se aplicará en la capa de aplicación y no dependerá de la configuración específica de servicios de los parámetros del servidor de red.

1. Replicación de datos programada

[0008] La presente invención define una manera en la que la transmisión de datos a través de una red inalámbrica con conmutación de paquetes puede programarse de manera inteligente, para mejorar el rendimiento del uso del ancho de banda de red sin afectar gravemente a la experiencia del usuario.

1.1 Cognima Replicate

[0009] El sistema Cognima Replicate está diseñado para replicar los datos del usuario entre un cliente móvil y un servidor de red, sin la intervención del usuario. Tanto el cliente como el servidor reconocen cuándo ha habido cambios en los datos para los que se necesita una replicación, de tal forma que se garantiza la integridad de la base de datos orientada a objetos distribuida que comparten el cliente y el servidor. Este enfoque crea una experiencia para los usuarios finales del sistema, de poder tener siempre acceso inmediato a todos sus datos, sin necesitar ninguna sincronización manual. Una consecuencia de hacer que la replicación sea invisible para el usuario consiste en que no se necesita ninguna indicación ni ningún control por parte del usuario sobre cuándo se produce la replicación. Los dispositivos del cliente y el servidor controlan cuándo debería producirse la replicación, y las decisiones de esos momentos se basan en parámetros dinámicos que puede configurar el operador de red móvil.

1.2 Replicación programada y operadores de red

[0010] Los operadores de red desean suavizar los puntos máximos y mínimos del ciclo de uso diario de la red con el fin de hacer un uso más eficiente del ancho de banda. Esto significa alejar el tráfico de datos de las horas punta y, en la medida de lo posible, desplazarlos a los puntos mínimos del ciclo. Los operadores valorarán la capacidad de modificar los ajustes que influyen en el momento en que se produce la replicación y, de esta manera, perfeccionar el rendimiento de la red.

[0011] Los operadores también desearán ofrecer servicios de distintos niveles de *QoS* y de coste (normalmente expresados como una elección de paquetes de nivel de servicios con su tarifa de datos actual) para satisfacer las distintas demandas de los consumidores. Esto es posible si se les da a los operadores de red la oportunidad de configurar, dinámicamente, los parámetros que definen la *QoS* de replicación desde la perspectiva de los usuarios.

[0012] Y lo que es más importante, los servicios que ofrecen los operadores deberían proporcionar a sus clientes una experiencia de usuario atractiva.

1.3 Replicación programada y expectativas del usuario

[0013] La tecnología Cognima presenta nuevos modelos mentales para los usuarios. Los modelos de programación de replicación, así como sus planes de servicios correspondientes, deben ser sencillos y coherentes para conseguir la aceptación del usuario. Los usuarios deberían estar protegidos de los detalles de replicación en la medida de lo posible. Los datos deberían replicarse de acuerdo con las expectativas del usuario.

[0014] Los usuarios quieren poder elegir un nivel de *QoS* entre una variedad de opciones: de lo contrario, consideran que están, bien pagando demasiado, bien que no están recibiendo un buen servicio. Las opciones de *QoS* para el usuario deberían ser sencillas. Los usuarios tendrán dificultades para valorar las ventajas relativas de un plan que ofrece *Contactos en 2 minutos, Fotos en 3 horas y Banca durante la noche* frente a uno que ofrece *Contactos inmediatamente, Fotos durante la noche y Banca en 30 minutos*, etc., por mucho que estas opciones puedan ajustarse a la investigación demográfica de los operadores de red.

[0015] Los usuarios se sentirán más cómodos si eligen un servicio estándar general; por ejemplo, básico y con la mejora quizás de un servicio particular. Los usuarios valorarán la oportunidad de mejorar temporalmente el intervalo de replicación, por objeto de servicio o individual a cambio de un coste. Por ejemplo, los usuarios pueden querer una opción premium "Enviar esta foto ahora", lo que invalidaría la prioridad por defecto con la que todas las demás fotografías se replican en el servidor de Cognima.

1.4 Replicación programada y arquitectura Cognima

[0016] Desde el punto de vista de la ingeniería, las soluciones deberían basarse en modelos sencillos en lugar de en conjuntos de reglas complejos. Las reglas que dependen de tiempos de espera (p. ej., *cada objeto debería tener su propio límite temporal, que empieza a contar hacia atrás cuando se introduce en el registro de cambios*) afectarán gravemente al rendimiento. De la misma manera, una solución que necesite que el CRE sondee el registro de cambios cada x segundos, también reducirá el rendimiento.

2. Implementación de *QoS*

2.1 Perfil de *QoS*

[0017] Introducimos el concepto de **perfil de *QoS***, que determina cuándo se replicarán los objetos de un determinado tipo para un usuario determinado.

[0018] El operador de red puede influir en el momento en que se replican todos los objetos de acuerdo con la tarifa en horas punta y la tarifa en horas valle o con los periodos de demanda de tráfico de red elevada. Un operador de red tiene la posibilidad de definir un perfil de temporización para cada aplicación, en base al cual cada dispositivo concilia las prioridades de replicación y los límites temporales de los objetos en el registro de cambios para determinar el comportamiento de la replicación. La forma de este gráfico será determinada por un número de factores, que incluyen la experiencia del operador de red en la monitorización de volúmenes de tráfico de datos.

[0019] El operador de red también puede influir en lo que se refiere a si deberían replicarse otros objetos (de menor prioridad) de un registro de cambios durante una conexión de datos abierta, una vez que se haya enviado el objeto que inicia la conexión. Es posible definir un **umbral de oportunismo** para controlar esto.

P. ej.: Hay diversos artículos no urgentes en el registro de cambios del dispositivo. El usuario cambia un ajuste del dispositivo que inicia de inmediato una conexión de datos. El operador de red ha especificado que cualquier otro objeto del registro de cambios por debajo de 20 kB debería replicarse con la conexión abierta.

[0020] Debería utilizarse un umbral de oportunismo diferente si el dispositivo está operando en una red en itinerancia, puesto que el coste para el usuario de iniciar conexiones puede sobrepasar el impacto para el operador de red de enviar más datos en periodos de hora punta. El umbral de oportunismo, junto con los demás parámetros de control de *QoS* en el dispositivo del cliente, son comunicados al cliente mediante la utilización del marco de replicación y, por lo tanto, se mantienen automáticamente en sincronización entre el cliente y el servidor.

[0021] En una implementación más avanzada del control de *QoS*, el servidor de red será capaz de determinar la carga de célula para un dispositivo móvil determinado. Cuando la carga de célula cae por debajo de un **umbral de carga de célula** definido, el servidor debería ser capaz de indicarle al cliente que puede iniciar la replicación oportunista. Esto proporciona más ventajas en las redes *all-IP* en las que el dispositivo del cliente tiene una dirección IP asignada de forma permanente y puede, de esta manera, ser contactado de manera inmediata por el servidor. De lo contrario, el proceso de enviar una solicitud de iniciación de comunicaciones al dispositivo del cliente influye en la carga de célula y añade una latencia al sistema que crea una oportunidad para que la carga de célula cambie antes de que pueda comenzar la replicación.

[0022] El operador de red puede actualizar el gráfico de tráfico de datos, el umbral de oportunismo y el umbral de carga de célula después de la implementación. Esto posibilita una optimización del control de *QoS* a la luz de la experiencia.

[0023] Todos los objetos que se unen al registro de cambios del dispositivo están sujetos actualmente al retraso **Pausa Antes de Conexión**. Sin embargo, como mecanismo definido de operador de red para definir hasta qué punto se agrupan por lotes los cambios, el retraso Pausa Antes de Conexión será sustituido en gran parte por el control de *QoS*.

2.2 Control de proveedor de servicios

[0024] El proveedor de servicios tiene la oportunidad de determinar un perfil de *QoS* para cada servicio de Cognima. Este perfil contiene conjuntos de límites temporales de replicación en los que se pretende que se repliquen los objetos de Cognima creados por ese servicio. El tiempo real en el que se pretende llevar a cabo la replicación en los límites temporales se determina mediante un número de elementos en el sistema que incluyen el estado del *software* del cliente de Cognima y el dispositivo del cliente, así como factores de red, tales como las horas punta y horas valle de la tarifa de datos, la carga de célula, etc. El proveedor de servicios puede establecer un límite temporal cero, es decir, solicitar la replicación inmediata para objetos de un determinado tipo. Si se trata de realizar la replicación y no se logra, se mantiene programada la replicación del objeto, pero sujeta al comportamiento de retroceso existente.

P. ej.: El proveedor especifica que la replicación de nuevos contactos en el servidor debería intentarse en un límite temporal de 2 horas. Si se crea un contacto durante un punto mínimo de la red o mientras la carga de célula es muy baja, el dispositivo puede enviar el contacto de inmediato. Si la temporización coincide con un punto máximo de la red o la carga de célula es alta, el dispositivo puede esperar que cambien las condiciones hasta 2 horas, pero pasado este tiempo debe intentar la replicación en cualquier caso.

[0025] El perfil de *QoS* define los siguientes factores como parámetros que determinan el momento de replicación:

- El tipo de objeto *(p. ej., contacto, foto, directorio DCC, protocolo de acceso DCC, etc.)*
- Cómo se ha creado el objeto *(p. ej., nuevo contacto en el teléfono, edición en contacto creada en el portal, etc.)*
- La dirección de viaje *(p. ej., las ediciones en contacto en el portal se producen de inmediato, las ediciones en contactos en el teléfono se producen en 2 horas)*
- El teléfono se encuentra en una red local/en itinerancia *(p. ej., la foto se replica de inmediato en casa, pero en 12 horas si es en itinerancia)*

[0026] Y opcionalmente...

- El tamaño del objeto *(p. ej., el contacto pesa más de 5 kB (es decir, contiene una imagen); enviar en 12 horas, el contacto pesa menos de 5 kB; enviar de inmediato)*
- Comportamiento diferente cuando se activa por primera vez el servicio (*p. ej., la subida inicial de fotos es inmediata; a partir de entonces, en 12 horas)*

[0027] El límite temporal para la replicación se le asigna a un objeto mediante un perfil de *QoS* basado en su momento de creación. Sin embargo, también es posible cambiar un límite ya asignado de acuerdo con eventos posteriores, tales como el cambio de la memoria disponible en el dispositivo o si el teléfono se mueve mediante itinerancia a otra red. Esto hace necesario que se recalculen de forma ocasional los pesos de los artículos en el registro de cambios.

[0028] El proveedor de servicios es capaz de definir un **periodo de validez** para los objetos del perfil de *QoS*. Si un objeto alcanza su periodo de validez mientras está en el registro de cambios, el objeto debería borrarse.

P. ej.: El proveedor de servicios especifica que una actualización del tiempo en el servicio DCC tiene un periodo de validez de 24 horas. Si el artículo no se ha replicado en un teléfono durante este tiempo, se borra del registro de cambios.

[0029] El proveedor de servicios puede definir un objeto como **sobrescribible**. Si un nuevo objeto se introduce en el registro de cambios y sustituye una versión anterior que todavía se encuentra en el registro de cambios, la versión anterior se borra. En esta situación, el proveedor de servicios puede determinar si el nuevo objeto debería adoptar la temporización del objeto que se ha sobrescrito o debería introducirse en el sistema con un nuevo límite temporal de replicación. La configuración predeterminada es que los nuevos objetos adopten las características de temporización de los objetos que sustituyen.

P. ej.: El proveedor de servicios especifica que un objetos de la clase de actualización del tiempo son sobrescribibles, así como que las nuevas entradas deberían adoptar la temporización de las que se sobrescriben. Esto significa que un mensaje de pronóstico antiguo es sobrescrito por uno nuevo, pero el nuevo no se refrena de replicarse.

[0030] Los proveedores de servicios pueden proporcionar distintas **clases de servicio** en el perfil programado. Una forma de realizarlo consiste en definir perfiles de *QoS* individuales para cada clase de servicio.

P. ej.: Los contactos en un servicio Oro se replican de inmediato después de su creación, pero con el servicio Plata en la misma red, la replicación puede tardar hasta 2 horas.

[0031] Se aplica una tarifa de *QoS* a un servicio de Cognima individual, aunque se le puede presentar al suscriptor como un conjunto de servicios a un precio particular.

[0032] El proveedor de servicios puede cambiar un perfil de *QoS* una vez se haya implementado; los cambios en el perfil se replican en los dispositivos del cliente pertinentes, de tal modo que la comprensión de cada perfil de *QoS* sea común entre todas las entidades del sistema.

2.3 Notas para derivar un perfil de *QoS*

[0033] Los parámetros de un perfil definido por un operador de red (o netop) deberían aspirar a compensar:

- Cuál es la experiencia de usuario aceptable para una determinada tarifa?
- Qué alcance debería proporcionarse para procesar por lotes los cambios (normalmente más procesamientos por lotes en itinerancia)
- Qué alcance debería proporcionarse para evitar los puntos máximos de red

[0034] Debería esperarse que los usuarios, por lo general, necesiten que los cambios realizados en un portal sean más rápidos que los realizados en el teléfono; deberíamos esperar que los usuarios que se encuentran en un PC tengan el teléfono consigo, pero no viceversa. La replicación inicial tras la activación debería ser inmediata para conseguir la mejor experiencia de usuario inicial.

[0035] El coste para el usuario de abrir diversas conexiones de datos en itinerancia probablemente sobrepasará el impacto para el operador de red de enviar más datos en periodos de hora punta.

3. Implementación algorítmica de *QoS*

3.1 Introducción

[0036] En el sistema de Cognima, se implementa *QoS* como incremento en la funcionalidad tanto del cliente como del servidor. En particular, requiere una modificación del comportamiento del registro de cambios e introduce un requisito para recalcular diversos atributos de entradas del registro de cambios en cola. La replicación puede producirse, entonces, como resultado del resultado de este recálculo.

3.2 Descripción del algoritmo

[0037] El algoritmo está formado por diversos componentes. Introducimos un **peso** de artículo de registro de cambios. Este peso indica la urgencia con la que ha de enviarse un artículo de registro de cambios; cuanto más pese, más urgente será el artículo. Introducimos un **umbral** de registro de cambios. Cualquier artículo que tenga un peso igual o superior al umbral ha de enviarse de inmediato. Cuando se realiza una conexión, se envían todos los artículos con un peso mayor que el umbral menos una delta. La delta representa el **oportunismo**.

[0038] Para mayor especificidad, decimos que el peso y el umbral pueden variar entre 0-100. El peso de un artículo que tenga que enviarse claramente ahora mismo es 100; un umbral de registro de cambios de cero indica que cualquier entrada en el registro de cambios puede replicarse inmediatamente.

[0039] Tanto el peso como el umbral pueden variar durante el día. Habrá variación predecible y también variación dinámica. Algunos ejemplos aclararán esto:

[0040] En la **figura 1**, la línea recta en el valor de peso 40 muestra que el peso permanece constante con el tiempo. El peso de un artículo que tiene que ir en un tiempo determinado se muestra en la **figura 2**. El peso del artículo empieza con un valor relativamente bajo, que indica prioridad baja y, a continuación, se dispara a medida que alcanzamos el límite temporal; el nuevo valor de 100 forzará al cliente a intentar la replicación del objeto en el momento T1.

[0041] Un artículo que debería ir solamente en un momento determinado se parecerá al diagrama de la **figura 2**, pero el salto viene en un momento determinado, no después de una duración particular. El peso de un artículo puede cambiar de forma dinámica. Digamos, por ejemplo, que el dispositivo empieza a quedarse sin espacio: el peso de los artículos podría aumentarse para sacarlos del registro de cambios y permitir la conversión en fantasma (véase Apéndice uno de la descripción actual para ver una explicación de este término). El umbral también tendrá un gráfico con el tiempo; el siguiente gráfico de la **figura 3** muestra un ejemplo de cómo puede aparecer un ciclo de umbral diario, con un umbral alto para protegerse del tráfico de valor bajo durante las horas punta (p. ej., después de las 9:00 durante varias horas) y un umbral inferior cuando las redes de datos están, tradicionalmente, más calmadas (entre las 00:00 y las 6:00, el umbral es cero).

[0042] El ejemplo, efectivamente, muestra un ciclo diario repartido en tres bandas tarifarias, quizás llamadas *horas valle*, *estándar* y *horas punta*, con el umbral de replicación establecido de manera adecuada para cada banda. Cabe observar que existe una cuarta banda justo después de la medianoche, en la que el umbral cae a cero; esto se introduce para asegurar que todos los registros de cambios se vacían una vez al día durante el periodo más tranquilo. Este periodo de umbral cero podría definirse una vez a la semana o en algún otro intervalo definido por el operador de red, pero se recomienda asegurar que el perfil de *QoS* definido no evita que algunos objetos se repliquen en absoluto. Al igual que para las otras bandas del perfil, el periodo de horas valle se prolonga hasta última hora de la tarde y hasta la noche y representa los momentos en los que cabe esperar que la red de datos experimente un tráfico bajo; la replicación de grandes objetos durante este momento permitirá al operador de red móvil hacer un uso mejor del ancho de banda limitado disponible durante las horas punta.

[0043] El periodo de umbral cero puede ser ajustado por el operador de red a la luz de la experiencia; pueden realizarse ajustes más finos para diferentes grupos de usuarios, por aplicación o incluso por usuario, de tal forma que se escalona el uso de red en horas valle y se asegura que la base de usuarios total para un servidor determinado no trate de conectarse al servidor de Cognima al mismo tiempo.

[0044] Se producirán cambios dinámicos en el umbral, por ejemplo, de la carga de célula o para apoyar un esfuerzo de comercialización que promocione un nuevo servicio (tiempo durante el cual puede ser preferible permitir los datos de tarifa básica en la red durante las horas punta para motivar el consumo). En caso de poder detectar el dispositivo que su célula no está ocupada, podría hacer descender su umbral un poco, lo que podría provocar una cierta replicación.

[0045] El núcleo del algoritmo es calcular el gráfico de umbral y el gráfico de peso de cada artículo del registro de cambios. Si el peso actual de cualquier artículo es superior al umbral actual, se establece una conexión. De lo contrario, se deduce la próxima vez que cualquier artículo sobrepase el umbral (es decir, cuando los gráficos se intersequen) y se programa un temporizador para este intervalo. Puesto que tanto los pesos como el umbral pueden ser dinámicos, hay distintos eventos que pueden provocar un recálculo:

- Se añade un nuevo artículo de registro de cambios
- El servidor puede insertar un nuevo valor de umbral al cliente. De hecho, este es un caso especial del evento anterior, puesto que el objeto de *QoS* en el cliente se controla mediante la replicación de manera normal, lo que significa que un nuevo valor de umbral será entregado al cliente situando el cambio en la cola del registro de cambios del dispositivo en el servidor; esta entrada de registro de cambios debe tener un peso de 100 para forzar la replicación inmediata y el cambio resultante de umbral puede provocar o retrasar la replicación de otras entradas que ya se encuentran en la cola.
- Un temporizador expira; será normalmente el temporizador el que indique el momento en el que está previsto que el peso de una entrada existente en el registro de cambios sobrepase el umbral de registro de cambios actual
- La carga de célula (o red) cambia
- La memoria disponible en el dispositivo del cliente cae por debajo de un determinado nivel
- El dispositivo detecta que su estado de itinerancia cambia
- Se implementa y se activa una nueva aplicación en el dispositivo
- Conexión terminada; esto también resulta en la creación / actualización del temporizador de la "próxima conexión".

[0046] Hay dos cálculos distintos: el gráfico de peso de un artículo y el gráfico de umbral.

[0047] Los parámetros que pueden afectar al peso de un artículo en un punto determinado en el tiempo son los siguientes:

- Dirección (cliente -> servidor o servidor -> cliente)
- Periodo de validez (normalmente codificado en la clase)
- Sobrescribible (normalmente codificado en la clase)
- Tamaño en *bytes*
- Hora introducida en el registro de cambios
- Prioridad
- Intervalo de tiempo de espera
- Tiempo asignado para la replicación
- Asignación de usuario de una prioridad no predeterminada a un objeto determinado (tal como la opción "enviar ahora" en una imagen para subirla a la cuenta de galería multimedia del usuario)
- Memoria disponible

[0048] El grado en que estos parámetros influyen en el peso es controlado por el proveedor de servicios (es decir, el operador de red) en el control, tal y como se describe en la sección "Control de proveedor de servicios" anterior.

[0049] Los parámetros que pueden afectar al valor de umbral actual de un registro de cambios son los siguientes:

- Momento del día
- Estado de itinerancia
- Carga de célula / red
- Tiempo transcurrido desde la última replicación
- Tarifa del usuario

[0050] Tras cada actualización del registro de cambios, el *software* del cliente también calcula el intervalo de tiempo para la siguiente intersección entre un gráfico de peso y el gráfico de umbral; esto se hace para que la replicación programada pueda producirse cuando sea necesario aunque no haya otros cambios antes de ese momento programado. Por lo general, el "siguiente evento" para el registro de cambios será, bien como ha predicho este cálculo de intersección, bien será creado por algún evento externo que sitúe una nueva entrada en el registro de cambios (que puede, evidentemente, forzar una actualización de los valores de todos los pesos y el umbral). Cabe observar que el evento de "siguiente conexión" puede presentar un valor de "nunca" si el registro de cambios está vacío, salvo que el perfil de *QoS* activo presente un periodo de umbral cero, como en el ejemplo anterior.

3.3 Bandas

[0051] Controlamos tanto el gráfico de peso del cliente como el gráfico de umbral mediante una estructura que llamamos **bandas**. Suponemos que todos los gráficos pueden describirse como conjuntos de líneas horizontales que saltan (concatenaciones de funciones escalonadas). Por lo tanto, el perfil de *QoS* para una determinada clase puede describirse mediante una matriz de **bandas**. Una banda se parametriza de la siguiente manera:

Parámetro	**Valores**	**Descripción**
Tipo de banda	Delta; absoluta	Si la banda se mide desde el momento de creación o en contraste con la hora del sistema.
Dirección	Servidor->cliente; cliente->servidor	Dirección de viaje del artículo.
Hora de inicio	Hora	Si tipo delta, una duración; si absoluta, hora de reloj.
Hora final	Hora	Si tipo delta, una duración; si absoluta, hora de reloj.
Peso	0-100	Urgencia de artículo

[0052] Las bandas pueden definirse como deltas desde una hora de inicio o en contraste con el reloj del sistema (absoluta). Un gráfico de peso para una clase que debería programarse para ir en 2 horas de la creación del objeto podría ser descrito por un par de bandas. La siguiente tabla podría describir el perfil de peso de *QoS* para la clase de Contactos.

Nombre de banda	Tipo de banda	Dirección	Hora de inicio	Hora final	Peso
Banda1	Delta	cliente->servidor	00:00	2:00	25
Banda2	Delta	cliente->servidor	2:00	24:00	90

[0053] Un gráfico de umbral que describe el gráfico en el ejemplo anterior se representaría de la siguiente manera:

Nombre de banda	Tipo de banda	Dirección	Hora de inicio	Hora final	Peso
Banda1	Absoluta	cliente->servidor	00:00	2:00	0
Banda2	Absoluta	cliente->servidor	2:00	7:00	20
Banda3	Absoluta	cliente->servidor	7:00	9:00	50
Banda4	Absoluta	cliente->servidor	9:00	11:00	80
Banda5	Absoluta	cliente->servidor	11:00	18:00	50
Banda6	Absoluta	cliente->servidor	18:00	24:00	20
Banda7	Absoluta	servidor->cliente	00:00	24:00	50

[0054] Cabe observar que hay un peso diferente para los objetos creados o modificados en el servidor y que este peso es constante a lo largo del ciclo diario. Este enfoque puede utilizarse para reflejar el hecho de que los cambios realizados por el servidor pueden tener una prioridad diferente que los que origina el cliente.

[0055] El cliente y el servidor soportan objetos de *QoS* que encapsulan las tablas anteriores e influyen en la programación de la replicación. Un dispositivo del usuario conservará un objeto de *QoS* para cada clase de datos instalada en el dispositivo, además de un único objeto de umbral que representa todo el sistema. Cada objeto de *QoS* contiene una matriz de bandas para describir un único gráfico de peso. Los puntos de intersección de estos gráficos determinan cuándo se replicarán los objetos de un determinado tipo; estos puntos de intersección deben calcularse determinando el peso de cada objeto y el valor de umbral para el sistema en un momento determinado.

3.4 Cálculo de peso del artículo de registro de cambios

[0056] Ha de calcularse el peso de un artículo en el registro de cambios. El objeto de *QoS* para la clase del artículo se recupera del almacén de datos. Se examina la estructura de bandas y se busca un peso del gráfico de peso (ya sea calculando la hora actual; hora creada en caso de un tipo de banda delta, ya sea comparando la hora del sistema en caso de un tipo de banda absoluta). Para incluir el tamaño del objeto como parámetro en el cálculo del peso, se han añadido dos campos más a la estructura de bandas.

Parámetro	Valores	Descripción
Límite de tamaño	Tamaño en *bytes*	Límite de tamaño de objeto por encima del cual se aplica el peso demasiado grande.
Peso demasiado grande	0-100	Peso de los objetos demasiado grandes.

[0057] Si el tamaño de objeto es superior al límite de tamaño, se utiliza el peso demasiado grande; cabe observar que este podría ser mayor o menor que el peso por defecto para la clase. En algunas aplicaciones, el

peso de objeto puede ser invalidado por una solicitud del usuario (efectivamente, un botón de "enviar ahora") que ajuste el peso en 100.

3.5 Cálculo del umbral de registro de cambios

[0058] El peso de umbral del registro de cambios actual puede extraerse del objeto de *QoS* de umbral mediante la utilización de la hora del sistema actual. A continuación, este valor puede modificarse mediante variables dinámicas; por ejemplo, si el dispositivo puede detectar el estado de itinerancia, esto puede influir en el umbral. Por lo general, el umbral será mayor en itinerancia, para reflejar el hecho de que la replicación será más cara; también es posible especificar un límite menor para el umbral en una banda determinada, evitando efectivamente que los objetos de muy baja prioridad se repliquen en absoluto antes de que el dispositivo vuelva a su red local. En implementaciones en las que el terminal móvil puede ser consciente de las condiciones de carga de célula locales, la carga de célula puede utilizarse como factor en el ajuste del valor de umbral actual: si la carga de tráfico de la célula local se encuentra por debajo de un valor (p. ej., 70 %) el umbral puede reducirse. Si la carga de célula se encuentra por encima de un valor (p. ej., 95 %) el valor puede aumentarse.

[0059] El objeto de *QoS* crea eventos de temporizador que representan los momentos del día en los que se sabe que el umbral cambia, que reflejan la forma del perfil de *QoS* tal y como define el operador de red. A medida que cada uno de estos eventos se dispara, se le asignará un nuevo valor al umbral de *QoS* y el recálculo de peso consiguiente permitirá que los objetos con el peso correcto se repliquen.

3.6 Cálculo del momento de la siguiente conexión

[0060] El gráfico de peso de un artículo ha de compararse con el gráfico de umbral del registro de cambios para encontrar el siguiente momento que el peso de artículo>= peso de umbral. Este cálculo ignorará el cambio dinámico del peso de umbral que es, por definición, impredecible. El cálculo de emparejar las bandas y comparar los pesos en momentos coherentes es relativamente sencillo.

[0061] Existen tres tipos de evento que podrían dar lugar a una sesión de replicación:

- un objeto de cambio ya en el registro de cambios puede aumentar su peso debido al desplazamiento a una nueva banda
- el umbral de *QoS* puede descender debido a un movimiento de una tarifa basada en el tiempo a otra
- una nueva entrada puede aparecer en el registro de cambios con un peso mayor que el umbral actual.

[0062] Para los dos primeros eventos, el momento en el que comienza la sesión es predecible y debe calcularse al final de cada sesión de replicación. Esto representa el siguiente momento de conexión programado.

APÉNDICE 1 DESCRIPCIÓN DEL SISTEMA DE REPLICACIÓN DE DATOS

[0063] La presente invención se describirá con referencia a una implementación de Cognima Limited de Londres, Reino Unido. Cognima ha desarrollado una tecnología de replicación de datos que se centra directamente en las necesidades de los proveedores de servicios móviles (MSP, por sus siglas en inglés) y operadores de red de aumentar la adopción por parte de los consumidores de servicios de datos, promover una lealtad mayor por parte de sus consumidores valiosos y diferenciar sus servicios de la competencia.

[0064] La solución de replicación de datos de Cognima aborda estos problemas al:

- Aumentar la adopción haciendo que los servicios de datos sean atractivos y se utilicen sin esfuerzo.
- Establecer una barrera elevada en relación con la tasa de cancelación mediante la realización de copias de seguridad de los datos personales de los suscriptores de forma segura en servidores controlados por los MSP de estos suscriptores.
- Permitir que el MSP cree servicios diferenciados controlando la experiencia del consumidor.

1. Resumen de los usos del marco de replicación de datos de Cognima

[0065] El marco de replicación de datos de Cognima permite que un proveedor de servicios móviles desarrolle servicios atractivos para mercados de consumidores. El MSP aloja un *servidor de Cognima* en su centro de datos. El servidor comprende una base de datos de Oracle, además del servidor de comunicaciones Java multiproceso de Cognima, alojados en un servidor de aplicaciones basado en estándares J2EE y *hardware* de Unix de alto rendimiento. En la sección 4 y en secciones posteriores, se describe la implementación técnica en detalle.

[0066] El marco de Cognima replica los datos introducidos en un teléfono móvil automáticamente (sin ninguna intervención por parte del usuario) en otros teléfonos por medio del servidor de Cognima. Asimismo, los datos de los sistemas externos conectados al servidor de Cognima se mantienen actualizados automáticamente en los teléfonos móviles.

[0067] Los suscriptores móviles que utilizan las aplicaciones habilitadas para Cognima experimentan una conexión ***instantánea, siempre disponible*** a su ***información personal*** y ***amigos***.

- La ***información personal*** puede incluir la libreta de direcciones del suscriptor, los mensajes, los datos bancarios, las cotizaciones bursátiles, los pedidos de pizza, el calendario, el tráfico actual en el camino al trabajo o cualquier otro contenido personal. Siempre se hace una copia de seguridad segura de los datos en el servidor de Cognima y se replican automáticamente en todos los dispositivos del cliente pertinentes.
- ***Siempre disponible*** significa que la información personal es accesible desde cualquier dispositivo o teléfono que lleve el suscriptor, esté conectado actualmente a la red o no, puesto que el usuario puede acceder siempre a la información personal almacenada localmente en el dispositivo). Los usuarios también pueden editar y gestionar sus datos personales directamente en el servidor a través de una interfaz web; el *teléfono virtual.*
- ***Instantáneo*** significa que los suscriptores no tienen que esperar para que se descarguen los datos de un servidor; la información más reciente está en sus teléfonos incluso antes de saber que la necesitan, puesto que esos datos se envían automáticamente al teléfono (p. ej., puede producirse un sondeo del teléfono; esto puede ser regular y periódico, tal como cada 30 minutos o en momentos predefinidos (4.00 p. m., 5.00 p. m., etc.). También puede producirse la inserción en el teléfono).
- Los suscriptores pueden compartir sus datos entre diversos dispositivos y con sus ***amigos***, puesto que el servidor de Cognima puede replicar estos datos en cualquier dispositivo definido o individuo definido.

1.1 Aplicaciones de Cognima ilustrativas

[0068]

Cliente	**Necesidad**	**Aplicación de Cognima**
Sarah	A Sarah le han robado el teléfono, incluidos algunos números de contacto y mensajes importantes, de los que no ha hecho ninguna copia de seguridad manual.	Cada vez que Sarah introduce datos en su teléfono, Cognima hace una copia de los mismos automáticamente en un servidor central en el centro de datos del MSP. Sarah puede comprarse un nuevo teléfono móvil y recuperar todos sus contactos y mensajes de forma instantánea del servidor central, siempre y cuando siga con el mismo MSP. También puede borrar sus datos del teléfono robado a través del portal del MSP.
Jill	Jill está de compras. Antes de hacer una compra cara, necesita saber si ha cobrado su salario en la cuenta bancaria. Sin embargo, se encuentra en el sótano de unos grandes almacenes y no tiene cobertura.	Cognima mantiene actualizado el contenido personal de Jill (incluidos sus datos bancarios) automáticamente en su teléfono móvil enviando de forma periódica (o en un momento predefinido o incluso se produce un cambio de inmediato) cualquier dato cambiado al móvil de Jill. Los datos más recientes se encuentran en el teléfono de Jill incluso antes de saber que los necesita. Puede acceder a ellos de forma instantánea, incluso sin cobertura.
Matthew	A Matthew le gusta mantener a sus amigos informados sobre su disponibilidad actual y su "estado de ánimo". También le gusta ver qué están haciendo sus amigos. Le interesa, principalmente, controlar qué está pasando en su grupo social y quiere hacerlo de un vistazo, sin tener que conectarse a la red o enviar una gran cantidad de mensajes caros.	Cognima comparte el perfil de presencia de Matthew con sus amigos. Cuando Matthew cambia su perfil (p. ej., selecciona un icono para indicar que tiene ganas de quedar) el icono se actualiza automáticamente en la entrada de la libreta de direcciones de Matthew del teléfono de sus amigos. Matthew puede ver información presencial de todos sus amigos de un vistazo en su propio teléfono. Incluso puede pedirle a su teléfono que le avise cuando un amigo tiene ganas de quedar o está aburrido, de tal forma que pueda llamar de inmediato.
Laura	Laura tiene dos teléfonos móviles; uno lo utiliza en el trabajo y el otro es un teléfono moderno que lleva por las tardes. Laura quiere mantener la misma libreta de direcciones en ambos dispositivos, pero odia introducir los datos dos	Cognima mantiene todos los datos del teléfono de Laura sincronizados automáticamente. Cada vez que edita datos en un teléfono, se replican de inmediato (o periódicamente o en un momento predefinido) en el servidor de Cognima que, a

	veces y nunca ha sabido cómo se utiliza el *software* de sincronización que venía con su teléfono. Cambiar la tarjeta SIM es engorroso y olvida datos en la memoria del teléfono.	continuación, actualiza su otro teléfono también. Nunca tiene que acordarse de pulsar un "botón de sincronización"; simplemente ocurre. Jill incluso comparte algunos de sus contactos de su teléfono con su marido, Geoff. Cuando Geoff introduce el número de móvil nuevo de su madre, se actualiza automáticamente en los teléfonos de Jill también.
Juha	Juha también tiene dos dispositivos móviles; un teléfono y una PDA habilitada para redes inalámbricas. Necesita leer y contestar a mensajes de correo electrónico y SMS en ambos dispositivos, pero se confunde y se frustra, y pierde productividad cuando su bandeja de entrada no está sincronizada.	Con Cognima, los SMS, los correos electrónicos y otros tipos de mensajes pueden leerse y enviarse desde cualquier dispositivo, así como con la utilización de una interfaz web de "*teléfono virtual*". Los mensaje se reciben en todos los dispositivos utilizados por el suscriptor y los mensajes enviados aparecen en la bandeja de salida de todos los dispositivos. Cualquier mensaje leído en un dispositivo, se marca como leído de forma instantánea en todos los demás dispositivos. Los mensajes borrados de un teléfono móvil pueden almacenarse y recuperarse a través del servidor de Cognima.

2. Ventajas para el suscriptor móvil

[0069] Cognima proporciona un marco ideal para implementar servicios de datos de consumo masivo basados en las siguientes ventajas principales:

- ***Facilidad de uso***: no se requiere ninguna intervención del usuario. Los suscriptores nunca han de pulsar un botón de "sincronización" o "descarga" para acceder a sus datos. La configuración del sistema y la transferencia de datos segura son completamente transparentes para el usuario final.
- ***Disponibilidad instantánea***: el usuario siempre puede interactuar de manera instantánea con los datos locales (incluso sin conexión a la red), mientras tienen lugar actualizaciones de forma silenciosa en segundo plano. Por ejemplo, los usuarios pueden leer su contenido personal cuando se encuentran en un tren subterráneo. La experiencia de usuario está separada de la transferencia de datos.
- ***Asequibilidad:*** El MSP puede controlar cuándo se produce la replicación, así como la calidad de servicio (*QoS*) proporcionada. Sin embargo, puesto que la experiencia de usuario está separada de la transferencia de datos, una *QoS* baja no afecta a la percepción del usuario del servicio. Significativamente, esto permite al MSP ofrecer servicios de bajo coste por suscripción con una *QoS* relativamente baja sin sacrificar la experiencia de usuario; por ejemplo, la replicación de datos puede producirse por la noche para servicios de datos que no son urgentes, tales como los estados bancarios, y aun así ser satisfactoria para los usuarios. La replicación de datos por la noche utiliza, por el contrario, ancho de banda infrautilizado y, por lo tanto, es mucho más barata que la replicación de datos en las horas punta. La replicación de datos urgentes (p. ej., información presencial) puede producirse en cualquier momento (con una inserción) periódicamente o (de forma opcional) continuamente y atraer un régimen de carga mayor. Asimismo, el uso eficiente de la memoria del teléfono y de la potencia del procesador permite que el *software* de cliente de Cognima se instale de manera económica incluso en los teléfonos más baratos de consumo masivo.

3. Ventajas para el proveedor de servicios móviles

[0070] Cognima proporciona a un MSP un medio para generar nuevos ingresos de datos, reducir la tasa de cancelación y diferenciar sus servicios de los de la competencia.

3.1 Mayor uso de los servicios móviles existentes

[0071] Cognima aumenta el uso de los servicios móviles existentes:

- Los servicios de mensajería y basados en contenidos cada vez son más convenientes e inmediatos y, por lo tanto, se utilizarán con más frecuencia.
- La inmediatez mejorada de la información presencial aumenta el uso del chat y la mensajería instantánea y una alerta cuando haya capacidad libre estimulará las llamadas de voz.
- La gestión sin esfuerzo de diversos dispositivos permite a los usuarios llevar un teléfono adecuado en cualquier ocasión y, por lo tanto, hacer más llamadas y enviar más mensajes.

3.2 Nuevos servicios atractivos

[0072] Cognima permite una introducción rápida de nuevos servicios de datos móviles atractivos y asequibles.

- Cognima proporciona una experiencia de usuario atractiva para nuevos servicios en teléfonos básicos utilizando solamente capacidad de red de repuesto. Esto es asequible y escalable para el operador de red, lo que le permite al MSP ofrecer unos precios comprensibles y predecibles para los suscriptores de consumo masivo.
- La mayoría del desarrollo de aplicaciones para los nuevos servicios de Cognima se produce por parte del servidor, lo que le permite al MSP llevar nuevos servicios al mercado rápidamente.
- El *software* del cliente de Cognima puede instalarse como una actualización de la memoria *flash*, dotando a los teléfonos de consumo masivo actuales de aptitudes similares a las de los teléfonos inteligentes. Pueden descargarse nuevas aplicaciones de *software* por el aire en teléfonos habilitados para Cognima actuales, lo que le permite a los MSP lanzar nuevos servicios de datos sin esperar que los nuevos dispositivos los soporten.
- Los desarrolladores de aplicaciones de terceros pueden hacer uso de la infraestructura de Cognima del MSP para desarrollar nuevas aplicaciones para la red del MSP.

3.3 Reducción de la tasa de cancelación

[0073] Los servicios de Cognima sirven de barrera importante contra la tasa de cancelación. Por ejemplo, un suscriptor que almacena su información personal de forma segura en el servidor de Cognima de su MSP puede comprarse un nuevo teléfono y recuperar, de inmediato, toda la información personal en su nuevo dispositivo. Toda esta información personal puede perderse si decide suscribirse a un proveedor de servicios diferente.

3.4 Diferenciación

[0074] Actualmente, los suscriptores tienen la misma experiencia básica de utilización de los servicios de datos móviles en todas las redes. Por ejemplo, la experiencia de utilizar servicios WAP está definida por los protocolos WAP, el navegador del teléfono y el contenido al que se ha accedido. Muchos MSP se han percatado de que deben distinguirse por proporcionar a sus suscriptores una experiencia de usuario única, pero unas limitaciones graves para personalizar los servicios en los teléfonos móviles les dificultan hacerlo.

[0075] Cognima proporciona a los MSP la capacidad de implementar servicios en el teléfono y, de esta manera, recobrar el control de la experiencia de usuario de sus suscriptores. Y lo que es más importante, Cognima hace posible esto sin sacrificar la interoperabilidad; la compatibilidad con los estándares del sector se consigue mediante la integración sencilla con el servidor de Cognima. La consecuencia directa es que la posición del MSP en la cadena de valor se refuerza frente a las marcas poderosas de fabricantes de teléfonos y de los proveedores de contenido.

4. Diseño funcional del marco de replicación de datos de Cognima

4.1 Introducción

[0076] En esta y en posteriores secciones de la descripción detallada, se pretende describir cómo funciona realmente el sistema de replicación de datos de Cognima. Se cubre el comportamiento de los dispositivos de los clientes, del servidor de Cognima y del cliente web, sin entrar en detalles de *hardware* específico, lenguaje de programación, diseño o entorno de clase de *software*. Se describen, en efecto, las estructuras de datos básicas y los algoritmos utilizados.

Términos

[0077]

Dispositivo del cliente	Teléfono, PDA u otro aparato que ejecutan el *software* de cliente de Cognima.
Servidor de Cognima	Servidor al que pueden acceder los dispositivos de los clientes que ejecuta el *software* del servidor de Cognima para replicar datos.
Replicación	Proceso de copiar datos de un dispositivo del cliente al servidor de Cognima y a otros dispositivos del cliente que pertenecen al mismo usuario.
Usuario	Ser humano que posee y utiliza al menos un dispositivo de cliente de Cognima.
Datos del usuario	Conjunto de información (contactos, mensajes, tonos de llamada, fotos, etc.) que un usuario puede querer almacenar y manipular en un dispositivo del cliente.

4.2 Objetivo

[0078] Los objetivos del *software* de Cognima son los siguientes:

- Permitir a un usuario acceso instantáneo para ver y modificar una copia "actualizada" de sus datos en diversos dispositivos portátiles que puedan conectarse a los datos de forma inalámbrica.

- Permitir a un usuario ver y modificar los mismos datos mediante la utilización de un navegador web tradicional.
- Proporcionar, fácilmente, copias de seguridad seguras de los datos de un usuario.
- Dar a un usuario una funcionalidad de datos potente en un teléfono barato desplazando el procesamiento caro y complicado a un servidor.

4.3 Descripción de nivel superior

[0079] Los dispositivos de los clientes guardan una copia de los datos del usuario en una base de datos del dispositivo del cliente. El usuario puede acceder a estos datos con o sin conexión de red y, por lo tanto, siempre tiene acceso al instante. Cuando un usuario cambia los datos del usuario en su dispositivo, los cambios se copian en un registro de cambios. El dispositivo del cliente se conecta periódicamente a un servidor de Cognima de la red inalámbrica para enviar los cambios desde el registro de cambios y recibir nuevos datos. Esto separa el acto de cambiar datos de la necesidad de conectarse a la red (es decir, la inserción no es continua en una implementación preferida). El servidor de Cognima actualiza su propia base de datos con los cambios de datos recibidos del dispositivo del cliente e ingresa datos en los registros de cambios de cualquier otro dispositivo que posea el cliente. La próxima vez que se conecten estos dispositivos, recibirán los cambios y, por lo tanto, los dispositivos se mantienen sincronizados, cada uno con una copia de los mismos datos.

[0080] El servidor de Cognima contiene un servidor web que posibilita que el usuario examine directamente, mediante la utilización de un navegador web, la copia de los datos guardados en la base de datos del servidor de Cognima y realice cambios en la misma tal como haría en un dispositivo del cliente. El servidor de Cognima también sirve como pasarela para que el usuario se comunique con otros servidores de la red/internet. Por ejemplo, el dispositivo del cliente puede, efectivamente, solicitar al servidor de Cognima el envío de un mensaje como SMS, correo electrónico o fax estableciendo unos cuantos indicadores en un objeto de mensaje y el servidor de Cognima contiene la funcionalidad para comunicarse con servidores de correo electrónico, servidores de SMS y aparatos de fax. Esto puede ampliarse a los servidores que guardan tonos de llamada, datos bancarios, juegos, etc. Es más fácil y económico construir el *software* en el servidor de Cognima para que hable con estos otros servidores que construir el *software* en el dispositivo del cliente.

5. Conceptos de nivel inferior

5.1 Estructuras de datos

5.1.1 Identificadores

[0081] Los datos del usuario de Cognima se describen mediante la utilización de la terminología de las bases de datos orientadas a objetos: clases y objetos. Desafortunadamente, hay lugar a confusión con conceptos de programación OO denominados de la misma manera, por lo que cabe tener precaución.

[0082] A todos los usuarios de una red de Cognima se les asigna un **identificador de usuario** (**ID**). Este ID es único en la red; es decir, proporcionado por un operador de red determinado. Todos los usuarios tienen una **dirección de Cognima**, que es una combinación de su ID de usuario y su dirección de URL del servidor de Cognima. Esta es única en el mundo. A cada dispositivo que pertenece a un usuario se le asigna un **identificador de dispositivo** (**ID de dispositivo**). El ID de dispositivo es único para el usuario. Es de sólo 8 bits, por lo que un usuario puede tener un máximo de 253 dispositivos (el ID 254 se reserva para la web, el ID 255 es de repuesto y el ID 0 es inválido). La totalidad de los datos del usuario se clasifican en clases (clase de contactos, clase de mensajes, clase de transacciones bancarias, etc.) y a las clases se les asigna un **identificador de clase** (**ID de clase**), que es único en el mundo. El ID de clase "12" se refiere a un contacto, por ejemplo.

[0083] Un ejemplo de una clase es un objeto, al que se le asigna un **identificador de objeto** (**ID de objeto**) único para el usuario; por ejemplo, un objeto de clase de contactos puede ser el contacto para "John Smith". El ID de objeto se genera mediante la concatenación del ID de dispositivo del dispositivo que creó el objeto con un recuento monotónico creciente que aumenta durante la vida del dispositivo. Por lo tanto, cada dispositivo puede crear un máximo de 16777215 objetos (si se alcanzara este límite, se podría restablecer el ID de dispositivo). Las clases se definen por las propiedades que las constituyen. Una clase es, fundamentalmente, una matriz de propiedades. Cada propiedad de la clase tiene un **identificador de propiedad** (**ID de propiedad**) que es único para la clase (y es, de hecho, justamente la posición de matriz de la propiedad en la matriz de propiedades, empezando desde cero).

5.1.2 Creación de objetos

[0084] Se crea un objeto en un dispositivo. Se le asigna un ID de objeto y se guarda en la base de datos del dispositivo. También se guarda una copia en un registro de cambios. La próxima vez que se conecte el dispositivo al servidor de Cognima, se envía la entrada del registro de cambios. El servidor de Cognima guarda el objeto en su base de datos (y registra la hora del sistema), realiza los procesamientos específicos de clase que

puedan necesitarse (tal como generar y enviar un correo electrónico) y añade entradas a los registros de cambios para cualquier otro dispositivo que pueda poseer el usuario y que haya presentado interés en la clase. (Las entradas deberían ser para la versión correcta de la clase en el dispositivo).

[0085] También puede crearse un objeto en el portal web. Se genera el ID de objeto (con la utilización del ID de dispositivo de 254, tal y como se describe anteriormente) y se procesa de la misma manera que el dispositivo. No hay ningún registro de cambios para el portal web; obtiene las selecciones directamente de la base de datos del servidor de Cognima.

[0086] Un objeto puede ser creado también por una aplicación de servidor (p. ej., un módulo de mensajería puede recibir un correo electrónico a partir del cual crea un objeto de mensaje). Se genera el ID de objeto (con la utilización del ID de dispositivo de 254, tal y como se describe anteriormente) y se procesa de la misma manera que el dispositivo.

5.1.3 Actualización de objetos

[0087] Una o varias propiedades de un objeto existente se modifican en un dispositivo. Los cambios se guardan en la base de datos del dispositivo. Cada propiedad cambiada se utiliza para generar una entrada en el registro de cambios del dispositivo. Estas son enviadas al servidor de Cognima.

[0088] Si la hora de la actualización es posterior a la hora del "último cambio" para la propiedad en la base de datos del servidor de Cognima, el servidor de Cognima guarda los cambios en su base de datos (y registra la nueva hora de "último cambio" para la propiedad), realiza los procesamientos específicos de clase requeridos y añade entradas a los registros de cambios para los demás dispositivos que pertenecen al usuario, que han declarado la clase y que tienen una versión de la clase que contiene la propiedad. La actualización también se sitúa en el registro de cambios para el dispositivo que originó el cambio. Esto puede parecer extraño, pero es necesario para gestionar la siguiente situación:

Un usuario tiene 2 dispositivos A y B. Actualiza la propiedad 7 en A sin conexión a las 5.00 p. m. y la actualiza en B sin conexión a las 6.00 p. m. Se conecta a la red con A primero. El valor de 7 en A se sitúa en el registro de cambios para ser enviado a B. Más tarde, se conecta B. Su valor de 7 es más reciente, por lo que el valor de 7 en B es enviado a, pero B obtiene el valor de A. La replicación del valor de 7 en B lo soluciona.

[0089] Si el servidor de Cognima recibe una actualización para un objeto que está marcado como borrado y la actualización es posterior al borrado, esto se interpreta como una cancelación de borrado. Se restaura el objeto, se actualiza y, a continuación, se sitúa una actualización del objeto en los registros de cambios para todos los dispositivos adecuados. Las actualizaciones del portal web o de las aplicaciones del servidor funcionan de la misma manera.

5.1.4 Borrado de objetos

[0090] Se borra un objeto del dispositivo. Se elimina de la base de datos del dispositivo y una entrada se sitúa en el registro de cambios anotando el ID de clase y el ID de objeto. La entrada es enviada al servidor de Cognima.

[0091] Si la hora del borrado es posterior a la hora de la última actualización del objeto, el servidor de Cognima marca el objeto como borrado en su base de datos, realiza cualquier procesamiento específico de clase y añade la entrada a otros dispositivos que pertenecen al usuario y han declarado la clase.

[0092] Si la hora del borrado es anterior a la hora de la última actualización, esto indica que el borrado es inválido y una actualización del objeto se sitúa en el registro de cambios para el dispositivo que originó el borrado.

[0093] El objeto borrado es visible en el portal web de una manera que aclara el estado borrado. El usuario puede seleccionar el objeto para la cancelación de borrado. La marca de borrado se elimina del objeto en la base de datos del servidor de Cognima y se sitúan entradas para actualizar el objeto en los registros de cambios para todos los dispositivos que pertenecen al usuario y han declarado la clase.

5.1.5 Tipos de propiedad

[0094] Cada propiedad tiene un tipo. Actualmente, hay 9 tipos de propiedad permitidos.

Nombre de tipo	Valor de tipo	Descripción de tipo
KcogTypeRef	0	ID de objeto de 4 bytes de otro objeto
KcogTypeInt	1	Valor entero de 4 bytes con signo
KcogTypeUInt	2	Valor entero de 4 bytes sin signo
KcogTypeFloat	3	Valor flotante de 4 bytes con signo

Nombre de tipo	Valor de tipo	Descripción de tipo
KcogTypeStr	4	CogString (entero sin signo de 4 bytes que conserva el número de caracteres en la cadena, seguido de los bytes de carácter)
KcogTypeTime	5	Valor entero de 4 bytes sin signo que indica el número de segundos desde la medianoche del 1 de enero de 1990
KcogTypeTypedStr	6	Valor entero de 4 bytes sin signo seguido de un CogString
KcogTypeBlob	7	Secuencia de bytes precedida de un entero sin signo de 4 bytes que conserva el número de bytes
KcogTypeArray	8	Estructura blob que puede contener una matriz de cualesquiera tipos de datos

[0095] Un **CogString** es un recuento de caracteres seguido de los caracteres. Si la cadena es ASCII, el espacio ocupado por la cadena será de (4 + recuento de caracteres) bytes. Si la cadena es Unicode, el espacio ocupado por la cadena será de (4 + (recuento de caracteres * 2)) bytes.

[0096] Un **CogTypedString** es un CogString precedido de un tipo (entero sin signo de 4 bytes). La única utilización de una cadena con tipo hasta el momento es un **Punto de Contacto**. El tipo identifica el tipo de punto de contacto (p. ej., dirección de correo electrónico, teléfono particular) y la cadena contiene la dirección (p. ej., bob@xxx.yyy, 01233556677).

[0097] Un **CogBlob** es una longitud en bytes seguida por ese número de bytes. Puede utilizarse para almacenar cualesquiera datos binarios.

[0098] Un **CogArray** se pasa como un "tipo" entero sin signo de 4 bytes, seguido de dos blobs. El "tipo" indica el tipo de elementos contenidos en la matriz. El primer blob es un **blob de índice**: contiene una secuencia de desplazamientos (enteros sin signo de 4 bytes) en el segundo blob. El segundo blob es el **blob de datos**, que contiene los elementos de la matriz como una secuencia de grupos binarios. Pueden extraerse elementos del blob de datos recontando el blob de índice para obtener el desplazamiento del inicio del elemento en el blob de datos. Esta es la estructura de secuencia de CogArray a medida que se pasa. Dentro de un sistema particular, puede aparecer como un vector convencional (es decir, ya analizado).

[0099] El único ejemplo implementado de un CogArray es **MessageAddress**. Cada elemento del MessageAddress es un AdressPair. Un **AddressPair** es un ID de contacto (ID de objeto de un objeto de contacto) seguido de un Punto de Contacto.

5.1.6 Parámetros de propiedad inteligentes

[0100] Algunas de las propiedades pueden transformarse en "inteligentes". Esto significa que pueden parametrizarse para un dispositivo específico para esculpir los datos en la propiedad para las características del dispositivo. En la práctica, los parámetros son dos enteros sin signo de 4 bytes; uno es un **tipo inteligente** y el otro es un **tamaño máximo**. Por ejemplo, la propiedad que contiene el cuerpo del texto de un mensaje puede parametrizarse en tipo inteligente kCogPlainText y tamaño máximo 100 en un teléfono económico con memoria limitada, pero parametrizarse para tener tipo inteligente kCogRichText y un tamaño máximo 1000 en una PDA con más memoria.

[0101] Los parámetros se almacenan en el servidor de Cognima cuando se añade la aplicación al dispositivo. Cuando se sitúan nuevos objetos o actualizaciones para esa clase en el registro de cambios del servidor de Cognima para ese dispositivo, se procesan de acuerdo con los parámetros inteligentes. Esto puede implicar, por ejemplo, truncar el texto, convertir el texto Unicode para estrechar el texto o convertir formatos de imagen.

[0102] Es importante para la integridad de los datos que el objeto guardado en la base de datos del servidor de Cognima sea una copia del objeto tal y como se generó. Incluso si se ve una versión más básica en un dispositivo, se puede manipular de forma efectiva la versión completa en el servidor de Cognima.

5.1.7 Versiones de clase

[0103] Tenemos el concepto de una **versión de clase** que está definida por un entero sin signo de 4 bytes. Una nueva versión de clase puede añadir propiedades al final de la clase antigua, pero puede no cambiar o eliminar propiedades existentes o insertar nuevas propiedades entre propiedades existentes. Esto debería permitir la interoperabilidad entre versiones. Las definiciones de clase con parámetros de propiedades inteligentes diferentes no son versiones diferentes.

5.2 Pasar los datos del usuario

[0104] Cognima utiliza la idea de **metadados de clase** para reducir los datos que han de copiarse entre bases de datos. Los metadatos de clase son, fundamentalmente, una matriz de **metadatos de propiedades**. Los

metadatos de propiedades son un ID de propiedad, un tipo de propiedad, un tipo inteligente y un tamaño máximo.

[0105] Los datos del usuario se transfieren como una secuencia solamente con la información de formato relativa a un ID de clase. Esta secuencia se analiza consultando los metadatos de clase. Por lo tanto, si se recibe una secuencia para la clase 6 y los metadatos de clase para la clase 6 dicen que la propiedad 0 es un KcogTypeUInt y la propiedad 1 es un KcogTypeStr, se sabe que los primeros 4 bytes de la secuencia deberían interpretarse como un entero sin signo, los siguientes 4 bytes deberían interpretarse como un entero sin signo que contiene el número de caracteres n en la cadena subsiguiente, los siguientes n (2 veces en caso de Unicode) bytes contienen los caracteres en la cadena, etc.

[0106] Los dispositivos del cliente anuncian al servidor de Cognima las clases que soportan. Esto permite que el dispositivo envíe, posteriormente, sólo datos del usuario sin procesar (con un encabezado que contiene el ID de clase, el ID de objeto y algunas cosas más) y, por lo tanto, reduce los requisitos de ancho de banda. Esto puede contrastarse, por ejemplo, con sistemas dependientes de XML que necesitan mucho más ancho de banda.

[0107] Las declaraciones de clase del dispositivo del cliente también contienen los parámetros de propiedades inteligentes, de tal forma que el servidor de Cognima pueda esculpir los datos para el dispositivo. Merece la pena hacer hincapié en que el *significado* de una propiedad se codifica de forma rígida en una aplicación. Los metadatos de clase manifiestan que la propiedad 2 de la clase 7 es una cadena con una longitud máxima de 30 caracteres. El código de la aplicación es el que interpreta la propiedad 2 de la clase 7 como el nombre de un equipo de fútbol.

5.2.1 Cuestiones de replicación de datos con más profundidad

[0108] Los datos se mantienen en objetos que se crean en dispositivos del cliente y en el servidor al que se conectan estos dispositivos (conocido como el servidor de Cognima). Estos objetos y sus posibles cambios se replican entre los dispositivos del cliente y el servidor de Cognima.

[0109] El diseño del proceso de replicación permite:

- Que se defina un conjunto de objetos que se replicarán, de tal forma que el mismo conjunto de objetos se mantendrá en un servidor de Cognima y todos los dispositivos del cliente que están conectados a ese servidor para un determinado usuario. Los nuevos objetos creados en cualquier dispositivo o en el servidor se replicarán en todos los demás dispositivos. Los cambios de cualquier propiedad de un objeto se replicarán en todos los dispositivos.
- Que solamente se transmitan los datos mínimos a través de la red para una determinada actualización, puesto que solamente los cambios en los datos se envían desde los clientes al servidor de Cognima o viceversa.
- Que una parte principal del diseño no necesitara guardar las horas de modificación de cada propiedad de un objeto en el dispositivo del cliente, puesto que la actualización de las mismas en los dispositivos del cliente limitados es lenta y guardar una hora de última modificación de cada propiedad en un objeto ocuparía demasiado espacio.
- En el servidor de Cognima, el almacenamiento de las horas de modificación de todas las propiedades de un objeto es excelente, puesto que el servidor tiene suficiente espacio de almacenamiento y potencia de procesamiento para gestionarlo.

5.2.2 Metadatos

[0110] Para que el sistema funcione, necesita una idea clara de qué propiedades están definidas para una determinada clase de objetos. Esto se realiza proporcionando al programador unos cuantos macros compiladores de C++ que permiten la definición de los metadatos de clase.

[0111] La definición de las propiedades que se han de utilizar en una clase da lugar a una definición de metadatos de clase. Esta definición le cuenta al CRE (motor de reconocimiento de Cognima) de qué tipo es una propiedad determinada y le permite empaquetar o desempaquetar un objeto o una propiedad para su transmisión a través de un enlace de datos. Para que el sistema CRE funcione, todos los clientes y el servidor deben tener la misma definición de metadatos de clase. Por lo tanto, ocurre lo siguiente:

- Cuando se manifiesta una nueva definición de metadatos en un dispositivo del cliente, se envía al servidor de Cognima y desde allí el servidor de Cognima la enviará a todos los demás clientes.
- Cuando se manifiesta una nueva definición de metadatos en un servidor de Cognima, la definición se envía a todos los dispositivos del cliente.
- Cuando un nuevo dispositivo del cliente se conecta a un servidor de Cognima por primera vez, todas las definiciones de metadatos se envían a ese dispositivo antes de que se envíe ningún objeto.

- En todos los casos anteriores, puede realizarse una optimización en el futuro, de tal forma que el servidor de Cognima solamente envíe la definición de metadatos a los clientes que accedan a la clase (y las propiedades específicas) a la que se refieren los metadatos.

5.2.3 Registro de cambios

[0112] El objetivo del registro de cambios es registrar cualquier cambio que se haya producido desde la última vez que se conectó el dispositivo del cliente al servidor de Cognima (o el servidor de Cognima al dispositivo del cliente). Mediante la utilización de las API de Cognima, las aplicaciones se conectan al CRE y pueden hacer que se creen o borren objetos o que se cambie una propiedad de un objeto. Estos cambios se añaden a un registro de cambios del dispositivo local a medida que se realizan junto con la hora en que se hizo el cambio. Se proporciona a los objetos identificadores únicos cuando se crean, de tal forma que un determinado objeto siempre pueda identificarse.

[0113] De la misma manera, la creación y el borrado de objetos y cambios en relación con las propiedades de los objetos por parte de las aplicaciones que se ejecutan en el servidor de Cognima, dan lugar a la adición de los cambios a todos los registros de cambios de la totalidad de los dispositivos del cliente registrados para ese usuario en el servidor de Cognima. La hora de los cambios se registra para cada objeto o propiedad.

[0114] Los registros de cambios pueden construirse de dos maneras:

- A medida que se crean los nuevos objetos y se cambian las propiedades (este es el caso, normalmente, para los dispositivos del cliente).
- O pueden construirse bajo demanda cuando se necesitan mediante la utilización de las horas de última modificación de los objetos y las propiedades si estos se almacenan en el sistema (en algunas circunstancias, este método puede utilizarse en el servidor de Cognima en lugar del método anterior).

5.2.4 Replicación

[0115] Cuando un dispositivo del cliente tiene artículos en su registro de cambios para enviar, se conectará al servidor de Cognima (y, de la misma manera, el servidor de Cognima se conectará al dispositivo del cliente). Por defecto, los artículos del registro de cambios se envían en el orden en que se añadieron al registro de cambios; sin embargo, pueden volver a establecerse prioridades de inmediato antes de enviarse para proporcionar servicios premium, datos urgentes, etc. Los artículos transferidos son los siguientes:

- Una definición de metadatos que incluye el tipo de cada propiedad de una determinada clase de objetos.
- Un nuevo objeto que ha sido creado; con los contenidos de las propiedades de ese objeto.
- Una propiedad ha sido cambiada; con el nuevo valor de la propiedad.
- Un objeto ha sido borrado.

[0116] En todos los casos anteriores, los ID adecuados son enviados para identificar el objeto, la clase y las propiedades implicadas. Todos los artículos del registro de cambios son marcados con la hora a la que se añadió el artículo al registro de cambios. Estas horas son siempre horas de máquina locales y se convierten en GMT mediante el enfoque de gestión del tiempo descrito en la sección 6.2.

[0117] Cuando un dispositivo del cliente recibe artículos del registro de cambios desde un servidor de Cognima:

- Cuando un dispositivo del cliente recibe un nuevo mensaje de objeto de un servidor de Cognima, añade este nuevo objeto a su base de datos local.
- Cuando un dispositivo del cliente recibe un mensaje de borrado de objeto de un servidor de Cognima, marca el objeto como borrado en su base de datos local.
- Cuando un dispositivo del cliente recibe un cambio de propiedad, siempre se da por sentado que el servidor de Cognima está autorizado sobre el estado actual de la base de datos y, por lo tanto, el cambio siempre se hace al valor de la propiedad mantenida en la base de datos local.

[0118] Un servidor de Cognima recibe artículos del registro de cambios desde un dispositivo del cliente:

- Cuando un servidor de Cognima recibe un nuevo objeto de un dispositivo del cliente, se añade a la base de datos del servidor de Cognima y también se añade a todos los registros de cambios de los dispositivos del cliente de ese usuario, aparte del registro de cambios de la máquina que envió el nuevo objeto en primer lugar.
- Cuando un servidor de Cognima recibe un borrado de objeto desde un dispositivo del cliente, el objeto se marca para su borrado y se añade un mensaje de borrado de objeto a todos los registros de cambios de los dispositivos registrados de ese usuario, aparte del registro de cambios de la máquina que envió el borrado de objeto en primer lugar.
- Cuando un servidor de Cognima recibe un cambio de propiedad, compara la hora del cambio con la hora actual conservada para esa propiedad en el servidor de Cognima. Si la hora del cambio de propiedad es

posterior a la conservada en el servidor de Cognima, el valor de propiedad cambia en la base de datos del servidor y este cambio también se añade a todos los registros de cambios de los dispositivos del cliente registrados de ese usuario - incluido el de la máquina que envió el cambio de propiedad (por si otra actualización de objeto ha sido enviada a esa máquina mientras tanto). Si el cambio de propiedad no era posterior al conservado en el servidor de Cognima, no se realiza ningún cambio, puesto que el valor de propiedad almacenado es más reciente, pero el valor se añade a la lista de valores de propiedad antiguos del servidor de Cognima, de tal forma que un usuario puede recuperarlo más tarde, si fuera necesario. Cuando se comparan las horas, se utiliza el enfoque de gestión del tiempo descrito en la sección 6.2, que aparece a continuación.

[0119] Cuando un dispositivo se conecta por primera vez a un servidor de Cognima, se le enviarán todas las definiciones de metadatos de clase y, a continuación, todos los objetos en la base de datos para ese usuario. Los mensajes de borrado, por lo general, solamente marcan un objeto para su borrado. La eliminación real del objeto de la base de datos puede producirse posteriormente, una vez se hayan borrado también todos los objetos relativos a ese objeto.

5.2.5 Optimizaciones

[0120] Una versión optimizada del protocolo de replicación anterior permite la agregación de las entradas en el registro de cambios. Si un registro de cambios (en el servidor de Cognima o en un dispositivo del cliente) aún no se ha replicado y se añade una entrada subsiguiente, las entradas existentes pueden analizarse para reducir potencialmente el número de entradas que necesitan replicarse durante la siguiente conexión:

- si la nueva entrada es una actualización de una propiedad que ya está programada para su actualización, solamente la última entrada necesita retenerse
- si la nueva entrada es un borrado de objeto, todas las actualizaciones de propiedad para ese objeto pueden eliminarse del registro de cambios
- si la nueva entrada es una orden de "restauración" y el borrado original todavía se encuentra en el registro de cambios, pueden eliminarse las dos entradas del registro de cambios

6. Algoritmos núcleo

6.1 Gestión del *endianness*

[0121] Los sistemas operativos son, básicamente, *little endian* o *big endian*, lo que consiste en elegir el orden de bytes en que se almacenan los números y las cadenas. Si dos ordenadores que tienen diferente *endianness* han de comunicarse, uno de los ordenadores tendrá que cambiar el *endianness* de sus paquetes de datos. En el entorno de Cognima, el *software* del cliente de Cognima utiliza el mismo *endianness* que el dispositivo del cliente anfitrión. El servidor de Cognima ha de determinar el *endianness* del dispositivo del cliente (utiliza un valor de referencia en el primer paquete de datos del cliente) y, a continuación, convertir los datos de entrada subsiguientes, si fuera necesario, para mantener un *endianness* coherente en el servidor de Cognima. El servidor de Cognima también ha de convertir cualesquiera datos de salida que devuelva al dispositivo del cliente.

6.2 Sincronización de las horas del sistema

[0122] De forma inevitable, diferentes dispositivos tendrán horas de sistema ligeramente diferentes. Los cambios que se envían desde los dispositivos del cliente al servidor de Cognima se marcan con la hora del sistema del dispositivo en el momento del cambio. Corresponde al servidor de Cognima decidir las horas en distintos dispositivos, de tal forma que pueda determinar el orden en el que tuvieron lugar los cambios y registrar la actualización correcta.

[0123] El inicio de sesión de un dispositivo contiene la hora del dispositivo actual. El servidor de Cognima debería ser capaz de compensar la latencia de la red y comparar la hora de inicio de sesión con su propia hora de sistema. Esto proporcionará una delta entre la hora del dispositivo y la hora del servidor de Cognima. Esta delta puede aplicarse a horas adicionales enviadas por el dispositivo en esa sesión.

[0124] El servidor de Cognima puede comparar deltas en sesiones sucesivas de un dispositivo para determinar el "desfase" del reloj en el dispositivo o los cambios de huso horario: no puede darse por sentado que todos los dispositivos del cliente en el sistema tienen relojes que están bien sincronizados entre sí:

- Las horas del reloj se desfasan en los dispositivos en función de la precisión del reloj del dispositivo.
- Por ejemplo, a algunos usuarios les gusta configurar los relojes 5 minutos más pronto.
- Algunos usuarios harán cambios en los relojes para representar la hora de verano en lugar de ajustar la configuración regional (y algunos OS pueden no proporcionar características regionales, lo que fuerza de todos modos al usuario a cambiar el reloj directamente).

[0125] Para resolver este problema, el servidor será responsable de ajustar las horas utilizadas por el dispositivo del cliente a GMT cuando se realicen comparaciones en el servidor y de GMT a la hora equivalente para el dispositivo del cliente cuando se envíen mensajes desde el servidor de Cognima al dispositivo del cliente.

[0126] El dispositivo del cliente etiquetará todos los artículos del registro de cambios con las horas obtenidas del reloj local; por lo que respecta al dispositivo del cliente, solamente trata con la hora a partir del propio reloj del dispositivo del cliente.

[0127] Cada vez que el dispositivo del cliente se conecta al servidor de Cognima, envía su perspectiva de la hora actual proporcionada por el reloj del dispositivo del cliente. A partir de esto, el servidor puede resolver:

- Cuál es la delta para GMT
- Si ha habido algún desfase en el reloj del dispositivo móvil desde la última vez que se conectó, puesto que el servidor guarda un registro de la última delta para GMT y de cuándo se realizó la última conexión y, por lo tanto, puede hacer una comparación. Si hay desfase, el servidor puede ajustar todas las horas enviadas por el dispositivo móvil proporcionalmente.

[0128] Por ejemplo, en la tabla que aparece a continuación, se muestra un patrón de los eventos con un dispositivo del cliente conectado a un servidor de Cognima. La hora del dispositivo del cliente es 5 minutos más lenta que el servidor de Cognima y suelta un minuto cada hora (un caso extremo para mostrar el punto). También para mostrar el punto, daremos por hecho que desde las 09:00 hasta las 12:00, el usuario está en un avión y no tiene contacto con el servidor de Cognima, por lo que no se conecta durante este tiempo:

Acción	Hora en el dispositivo del cliente	Hora en el servidor de Cognima (GMT)
El dispositivo del cliente se conecta al servidor de Cognima	09:00	09:05
Se hace un cambio en la propiedad A	10:00	X
Se hace un cambio en la propiedad B	11:00	Y
El dispositivo del cliente se conecta al servidor de Cognima	12:00	12:08

[0129] Con el fin de averiguar si los cambios de propiedad se realizaron antes o después de la hora almacenada en el servidor de Cognima, han de averiguarse las horas X e Y. A partir de la información anterior, el servidor de Cognima sabe que la última vez que se conectó el cliente era hace unas 3 horas y, en ese momento, la diferencia temporal era de 5 minutos, mientras que ahora es de 8 minutos. Por lo tanto, si suponemos que el desfase del reloj se produce de forma lineal, el servidor de Cognima puede averiguar que el dispositivo está 5 minutos por detrás de GMT y que el reloj se desfasa hacia atrás un minuto cada hora.

[0130] A partir de esto, es posible averiguar que la hora que el dispositivo del cliente conoce como 10:00 para el cambio de la propiedad A necesita que se le añadan 5 minutos para el desfase inicial, más un minuto para el desfase adicional que se produjo en la hora hasta que se cambió esa propiedad.

[0131] Del mismo modo, necesita ajustarse la propiedad B a las 11:07; el desfase inicial de 5 minutos más 2 minutos desde que transcurrieron dos horas desde las 09:00 hasta las 11:00 cuando se cambió la propiedad.

[0132] En la práctica, la delta de la hora entre la hora del dispositivo del cliente y GMT puede ser minutos, pero el desfase será del orden de fracciones de segundo por hora.

6.2.1 Ajustes horarios

[0133] Además de la delta de GMT y de cualquier desfase en el reloj del dispositivo del cliente, los usuarios también pueden cambiar la hora del dispositivo del cliente. Pueden hacer esto para reajustar la hora a la hora local actual (podemos dar al usuario la opción de que esto ocurra automáticamente pero algunos usuarios pueden querer mantener su propio control de la hora de su dispositivo de cliente; p. ej., les gusta ajustar el reloj 5 minutos más rápido). También pueden hacer ajustes para reflejar un cambio de la hora local (es decir, horario de verano o huso horario cambiante). El objetivo es que el usuario pueda cambiar el reloj en el dispositivo a cualquier hora que le venga bien al usuario y el dispositivo simplemente tiene esto en cuenta.

[0134] Cuando el usuario realiza un cambio en la hora del dispositivo del cliente, la mayoría de los sistemas operativos informarán de este cambio (para sistemas que no hagan esto, puede sondearse la hora, por ejemplo, cada minuto para comprobar dicho cambio). Cuando detecta un cambio de la hora el dispositivo del cliente resolverá la delta entre la nueva hora y la hora que era antes del cambio. Por ejemplo, esto puede ser un cambio de más una hora a medida que un usuario cambia de huso horario. El dispositivo del cliente almacena esta diferencia horaria como la hora de ajuste, que guarda para la próxima conexión al servidor de Cognima. El dispositivo del cliente también pasa por cada entrada del registro de cambios y actualiza todas las horas en el

registro mediante la hora de ajuste. Esto garantiza que las entradas en el registro de cambios siempre sean relativas a la hora local del dispositivo del cliente.

[0135] Varios de estos cambios podrían hacerse entre conexiones al servidor de Cognima; cada vez que la cantidad del cambio de hora se añade a la hora de ajuste y se actualiza el registro de cambios, de tal forma que las horas del registro sean relativas a la hora local del dispositivo del cliente.

[0136] La próxima vez que el dispositivo del cliente se conecta al servidor de Cognima, el dispositivo del cliente envía, con el inicio de sesión, la hora de ajuste almacenada; es decir, la cantidad mediante la cual el reloj del dispositivo del cliente se ha ajustado hacia atrás o hacia delante desde la última conexión. El servidor de Cognima puede, entonces, eliminar esta cantidad de la hora de la delta a GMT y el cálculo de desfase.

6.2.2 GMT a dispositivo del cliente

[0137] El mismo conjunto de cálculos puede realizarse en orden inverso para convertir las horas GMT de los cambios realizados en el servidor de Cognima en la hora local correcta para un determinado dispositivo del cliente.

6.3 Adición de una aplicación

[0138] Una aplicación utilizará una o más clases para conservar los datos del usuario. La definición de la clase se codifica de forma rígida en la aplicación. La versión de la clase se coordina mediante las versiones de la aplicación.

[0139] Pongamos por caso que una aplicación de estadísticas utiliza una clase Futbolista para conservar datos sobre futbolistas. Cuando la aplicación se inicia en un dispositivo del cliente por primera vez, le pregunta al dispositivo qué versión de la clase Futbolista posee ya el dispositivo. Si la versión del dispositivo es la misma que la versión que se ha codificado de forma rígida en la aplicación para su uso, no ha de hacerse nada más.

[0140] Si el dispositivo posee una versión más reciente de la clase Futbolista, se necesita que la aplicación sea lo suficientemente robusta para gestionar más propiedades de las que esperaba. (Esta situación se produciría si se tuviera una clase utilizada por diversas aplicaciones y, por algún motivo, se instalara una versión más antigua de una de las aplicaciones. Esto debería ser excepcional: lo ideal sería que las aplicaciones interdependientes se actualizaran conjuntamente).

[0141] Si el dispositivo posee una versión más antigua de la clase Futbolista (o ninguna versión), la versión de la aplicación de la clase Futbolista debería sustituirla. La nueva versión se envía al servidor de Cognima. El servidor de Cognima, por lo tanto, mantiene una lista de las versiones de las clases utilizadas en todos los dispositivos.

[0142] Las páginas del portal web serán el equivalente de la aplicación del dispositivo codificada de forma rígida. La web puede extraer objetos de la base de datos de acuerdo con la versión más reciente de la clase y, si hay más propiedades de las que se esperaba con la codificación rígida, puede ignorarlas. Por lo tanto, la web no necesita manifestar las versiones de clase.

6.4 Optimización del registro de cambios

[0143] El servidor de Cognima mantiene registros de cambios para todos los dispositivos que registren cambios que serán enviados a los dispositivos la próxima vez que los dispositivos se conecten. Habrá optimizaciones que pueden realizarse en el registro de cambios, por ejemplo:

- Si>2 actualizaciones para la misma propiedad están en cola en el registro de cambios, solamente la última ha de guardarse.
- Si un borrado está en cola para un objeto, cualquier actualización por delante en la cola puede eliminarse.
- Si una actualización está en cola para un objeto, cualquier borrado por delante en la cola debería eliminarse.
- Si un dispositivo registra una nueva aplicación, podría haber potencialmente muchos objetos para ser enviados a la misma (p. ej. historial de mensajes). El registro de cambios debería tener solamente un número *razonable* de objetos añadidos al mismo (p. ej., los 20 mensajes más recientes).

7. Conversión en fantasma, resurrección, anclaje y retirada

[0144] El espacio disponible en un dispositivo del cliente para conservar los datos del usuario serán, normalmente, órdenes de magnitud inferiores al espacio disponible en el servidor. El dispositivo necesita conservar un subconjunto de datos y el usuario debería tener que hacer el mínimo trabajo posible para mantener este subconjunto. La conversión en fantasma y la retirada son herramientas que ayudan a hacer esto.

[0145] Una definición de clase puede incluir indicar determinadas propiedades como "convertibles en fantasma". Esto significa que si el objeto se convierte en fantasma, esas propiedades se anularán y se liberará espacio en el dispositivo del cliente. La conversión en fantasma se hace automáticamente en el dispositivo. La decisión sobre qué objetos se han de convertir en fantasma se hace siguiendo una "regla de conversión en fantasma" y aplicando la regla cada vez que se crea o se actualiza un objeto. La regla define el número máximo de una selección de objetos. Cuando se sobrepasa el máximo, los objetos de la selección al final de un orden de clasificación se convierten en fantasma.

[0146] Por ejemplo, la clase puede ser mensajes, la selección puede podría ser mensajes en la bandeja de entrada, el orden de clasificación podría ser por fecha/hora y el número máximo podría ser 50. Si hay 50 mensajes en la bandeja de entrada y llega un nuevo mensaje, el mensaje más antiguo de la bandeja de entrada se convierte en fantasma. Mediante la conversión en fantasma se puede eliminar el cuerpo del mensaje pero dejar suficiente información de encabezado para que se identifique el mensaje.

[0147] La retirada (también conocida en el pasado como autoborrado y eliminación) es similar a la conversión en fantasma pero funciona mediante la eliminación del objeto completo, no sólo de una parte del mismo.

[0148] Ni la conversión en fantasma ni la retirada son notificadas al servidor de Cognima. Son estrictamente locales para el dispositivo del cliente. Por lo tanto, diferentes dispositivos pueden tener diferentes números de objetos. Los datos de los dispositivos están, todavía, fundamentalmente sincronizados, pero los dispositivos conservan diferentes subconjuntos de datos.

[0149] Si el usuario desea resucitar a un fantasma, se pasa una solicitud desde el cliente al servidor de Cognima para resucitar el objeto. Se envía una actualización del objeto al dispositivo y el objeto vuelve a su estado normal.

[0150] Los objetos individuales pueden anclarse. Un objeto anclado nunca se convierte en fantasma ni se elimina. El anclaje puede ser elegido por el usuario o puede producirse automáticamente. Por ejemplo, un objeto que se ha resucitado, se ancla automáticamente.

8. Replicación de usuario - compartir objetos

[0151] Hay muchas aplicaciones para las que se contempla que será útil que los usuarios puedan compartir objetos. La manera general en la que funcionará esto es la siguiente: un usuario necesita saber la dirección de Cognima de los usuarios con los que puede querer compartir objetos. Es más adecuado analizar la recuperación de estas direcciones en detalle en la arquitectura del servidor de Cognima. Aquí, damos por sentado que dicha lista está disponible.

[0152] Un conjunto de una o varias direcciones de Cognima está unido al objeto que se ha de compartir. Puede configurarse el objeto para su lectura solamente (para que la gente con la que se comparte, no pueda modificarlo). Cuando el servidor de Cognima recibe el nuevo objeto (o recibe una actualización para el mismo) desde la web o un dispositivo del cliente, lo replica como normal.

[0153] También consulta la lista de direcciones de Cognima de personas con las que se comparte. Marca el objeto con un ID de creador (es decir, la dirección de Cognima del propietario del objeto + el ID de objeto) y lo envía a las personas con las que se comparte. Los usuarios con los que se comparte pueden estar situados en el mismo servidor de Cognima o encontrarse en diferentes servidores de Cognima. El servidor de Cognima de las personas con las que se comparte recibe el objeto. Si es un objeto nuevo, asigna un nuevo ID de objeto (y toma nota del ID de creador). Si es una actualización, encuentra el objeto mediante la utilización de un ID de creador.

[0154] Si la persona con la que se comparte tiene permiso para actualizar el objeto, la actualización puede volver a replicarse en el propietario del objeto mediante la utilización del ID de creador.

9. Visualización de datos

[0155] Los dispositivos pequeños convencionales como una PDA tienden a tener sistemas de archivo sencillos que permiten que las aplicaciones lean y escriban datos en algún tipo de almacenamiento que conservará los datos cuando la aplicación no está en ejecución. Por lo general, estos programas tenderán a leer en el conjunto de datos disponible y, entonces, proporcionarán una interfaz de usuario para visualizar los datos en la pantalla. Esto presenta algunos inconvenientes:

- Leer en los datos cuando el programa se inicia requiere tiempo
- La aplicación necesita almacenar todos o algunos de los datos en la memoria, lo que significa que ahora ocupa más memoria en el dispositivo del cliente
- Permitir que más de una aplicación pueda acceder al mismo conjunto de datos se convierte en no trivial
- Código similar para leer y manipular los datos aparece en diversas aplicaciones que se ejecutan en el dispositivo

[0156] El enfoque de Cognima es diferente:

- Los datos se almacenan en una base de datos orientada a objetos a la que puede accederse mediante diversas aplicaciones
- Una aplicación de Cognima no lee en todos los datos con los que trata de una base de datos. En su lugar, crea una *selección*; un subconjunto de los datos en los que está actualmente interesado. En general, esta selección coincide con los datos que se están visualizando actualmente en la pantalla de los dispositivos. Por lo tanto, solamente los datos que está utilizando la aplicación actualmente se conservan en la memoria, de tal forma que se ahorra mucho espacio de memoria.
- Todo el trabajo de almacenamiento, clasificación e indexación de los datos lo realiza una base de datos orientada a objetos y, por lo tanto, esta funcionalidad no ha de repetirse en cada aplicación.
- Cuando han de hacerse cambios en los datos de una aplicación, la aplicación nunca actualiza directamente su propia visualización de los datos. Los cambios actualizarán las propiedades de un objeto o crearán o borrarán un objeto. Un cambio en los datos podría ser materializado por otra aplicación o una actualización recibida de un servidor de Cognima debido al cambio de datos en otra máquina.
- Cuando una aplicación establece una selección, proporciona una lista de criterios en función de los que los datos, bien se incluyen en la selección, bien se excluyen de la misma; debido a esto, el motor de replicación de Cognima puede decir a qué aplicaciones se les notificará la creación, el borrado o la actualización de un objeto.
- Cuando ha de enviarse una actualización a la aplicación, se llama al código de la aplicación relacionado con la selección que contiene estos datos y, de esta manera, la aplicación puede responder a los cambios que se han llevado a cabo.
- Cuando se establecen las selecciones, la aplicación también puede especificar cómo se clasifican los datos y si sólo se necesita una pequeña ventana de la lista de datos clasificados (conocido como vista).

[0157] Este enfoque es similar al enfoque de volver a dibujar la pantalla, utilizado para volver a dibujar pantallas de gráficos en los sistemas de ventanas. Cuando una zona de la pantalla ha de volverse a dibujar, la aplicación que es responsable de ese pedazo de pantalla es llamada para volver a dibujar la pantalla.

9.1 Ejemplo

[0158] Un dispositivo del cliente puede tener una aplicación de contactos en ejecución. Este dispositivo replica los datos con un servidor de Cognima conectado a otros dispositivos del cliente que también tienen una aplicación de contactos en ejecución. Se define una clase de objeto para un contacto que contiene nombres y números de teléfono y estos se replican en todos los dispositivos de un determinado usuario.

[0159] Una aplicación de un dispositivo puede tener una visualización que muestra todos los contactos por la letra de inicio; por ejemplo, la interfaz permite que el usuario pulse un botón D para mostrar todos los nombres que empiecen por la D. Esta aplicación establecerá una selección que contiene objetos:

- Cuando la clase se defina como contactos
- Cuando el nombre empiece por la letra seleccionada (p. ej., D)

[0160] Cuando se define la selección, la aplicación también define el código que llamará el CRE cuando se añadan, se borren o se actualicen los objetos. La primera vez que se establezca la selección, este código volverá a ser llamado con el primer conjunto de objetos que satisfagan los criterios anteriores.

[0161] Si se le pidiera a la aplicación crear un nuevo contacto con un nombre que empieza por la D, la aplicación crearía el objeto y nada más. El CRE detectaría el nuevo objeto y volvería a llamar al código de selección para notificarle el nuevo objeto.

[0162] De la misma manera, si se creara un nuevo objeto de contacto en otro dispositivo y se replicara en el dispositivo del cliente, si el nombre de ese contacto empezara por la D, se le notificaría a la aplicación.

9.2 Clasificación

[0163] Los datos de las selecciones, por lo general, han de clasificarse; a menudo, de tal forma que cuando se visualizan, los usuarios pueden ver los datos en un formato lógico. Cuando se define una selección, el orden de clasificación puede especificarse: las propiedades que se han de ordenar, en qué orden y qué algoritmos de clasificación se han de utilizar.

9.3 Vistas

[0164] Pueden haber muchos artículos de datos en una selección. Normalmente, cuando se visualizan los datos, puede que no todos quepan en la pantalla y, por lo tanto, el usuario necesitará desplazarse hacia arriba y hacia abajo de los datos. Una vista proporciona esta funcionalidad mediante la especificación del número de artículos de datos con los que quiere tratar la selección y el número del primer artículo de datos fuera de la lista completa de datos que la aplicación quiere que aparezcan en la selección.

[0165] Las vistas son importantes porque permiten que una aplicación pueda limitar la cantidad de datos que almacena localmente a solamente la cantidad necesitada para visualizar en la pantalla esto, de modo que se reduce la duplicación de datos innecesaria.

9.4 Eficacia

[0166] Cognima ha llevado a cabo algunas optimizaciones de eficacia sobre cómo se transfieren los datos entre el servidor de Cognima y la aplicación del cliente; cuando se hacen diversos cambios de datos, los datos son enviados en bloques y se informa a la aplicación de que los cambios se han completado, de tal forma que la aplicación solamente necesita actualizar su interfaz de usuario una sola vez.

9.5 Ejemplo

[0167] A modo de ejemplo, definiremos una selección llamada ContactSelection. Este es el código que el marco volverá a llamar cada vez que se haga un cambio en cualquiera de los objetos seleccionados. En el marco de Cognima, esto se implementa como un objeto que se deriva de la clase con modelo CObdSelect - especificando el tipo de objeto que se quiere tener en la selección como argumento de modelo.

```
class CContactSelect : public CObdSelect<CContact>
 {
public:
 CContactSelect (CObd *aObd);
 void ObjectAdded (CContact *aObject);
 void ObjectUpdated (CContact *aObject);
 void ObjectRemoved (const TObdObjectId aObjectId);
private:
  bool ListContacts();
};
```

[0168] Los métodos ObjectAdded(), ObjectUpdated() y ObjectRemoved() son llamados por el marco cada vez que, respectivamente, se añade, actualiza o elimina un objeto. Cuando se implementa la clase selección, no se necesita implementar todos estos métodos si no se desea adoptar medidas de ejemplo sobre cualquiera de estos eventos; en algunos casos, se puede establecer una selección para mantener una lista de un determinado conjunto de objetos, pero solamente comprobar esa lista sobre algún otro evento y, por lo tanto, los métodos anteriores no se necesitarían.

[0169] Hemos definido un método privado adicional llamado ListContacts() - este registrará todos los contactos actuales conservados por la selección.

[0170] Aquí está la implementación de esta clase:

```
CContactSelect::CContactSelect(CObd *aObd)
 : CObdSelect<CContact>(aObd)
 {
 }
void CContactSelect::ObjectAdded(CTestContact *aContact)

  ObdLog(ObdLogApp,L"New contact added:" <<aContact->GetName());
  ListContacts();
 }
void CContactSelect::ObjectUpdated(CTestContact *aContact)
 {
  ObdLog(ObdLogApp,L"Contact updated:" <<aContact->GetName());
  ListContacts();
 }
void CContactSelect::ObjectRemoved(const TObdObjectId aObjectId)
 {
  ObdLog(ObdLogApp,L"Contact deleted - Id:" <<aObjectId);
  ListContacts();
 }
void CContactSelecr::ListContacts()
 {
  ObdLog(ObdLogApp,L"Contacts list:");
  for (unsigned long Index=0; Index<iResult.size(); Index++)
  {
        CTestContact*Contact=iResult[Index];
        ObdLog(ObdLogApp,Contact->GetName() <<","
              <<Contact->GetPhone() <<","
              <<Contact->GetEmail());
```

```
    }
}
```

[0171] El constructor simplemente llama al constructor CObdSelect por defecto. Los métodos ObjectAdded(), Updated() y Removed() imprimen qué cambio se hizo y, a continuación, llaman a ListContacts() para mostrar cuál es el contenido actual de la lista.

[0172] ListContacts() muestra cómo puede accederse a la lista actual de objeto conservada por la selección. La lista actual de punteros a objetos se conserva en una clase contenedor llamada iResult; a esta puede accederse a continuación mediante integradores de clase contenedor normales. En este caso, simplemente se examina la lista y se imprimen todos los objetos de la lista.

REIVINDICACIONES

1. Método de replicación automática de objetos de datos entre un dispositivo móvil y un servidor, conectados a través de una red inalámbrica, en el que la temporización de la replicación de datos a través de la red es determinada por un operador de red que aplica parámetros que hacen un uso eficiente del ancho de banda de red; donde el método comprende las etapas de:

(i) listar en un registro de cambios todos los objetos en el dispositivo y/o el servidor que se han de replicar;
(ii) definir los parámetros mediante la utilización de (a) un peso asociado a cada objeto que define la urgencia con la que ha de replicarse dicho objeto y (b) un umbral que depende del tiempo, en el que el umbral varía con el tiempo de tal forma que se hace un uso eficiente del ancho de banda disponible;
(iii) comparar localmente el peso de cada objeto con el umbral en un momento determinado, determinando el resultado de la comparación si se envía el objeto para la replicación o no en ese momento.

donde los criterios que son pertinentes en lo que se refiere a la urgencia con la que ha de replicarse un objeto están representados por dicho peso asociado a ese objeto.

2. Método de acuerdo con la reivindicación 1, en el que se establece una conexión en un momento determinado si el peso de un objeto sobrepasa el umbral en ese momento.

3. Método de acuerdo con la reivindicación 1, en el que el peso de un objeto en un momento determinado depende de uno o varios de los siguientes elementos:

(a) La dirección de replicación de datos (dispositivo a servidor o servidor a dispositivo)
(b) El periodo de validez, que define el tiempo o la duración después de la que el objeto será borrado automáticamente si todavía está presente en el registro de cambios
(c) Si el objeto se puede sobrescribir o no
(d) El tamaño en *bytes*
(e) La hora introducida en el registro de cambios
(f) La prioridad
(g) El intervalo de tiempo de espera
(h) El tiempo asignado para la replicación
(i) La asignación de usuario de una prioridad no predeterminada a un objeto determinado
(j) La memoria disponible.

4. Método de acuerdo con la reivindicación 1, en el que el operador de red puede provocar la modificación del peso de un objeto en cualquier momento.

5. Método de acuerdo con la reivindicación 1, en el que el operador de red puede provocar la modificación del umbral en cualquier momento.

6. Método de acuerdo con la reivindicación 1, en el que el umbral puede variar con el tiempo de manera diferente en distintos grupos de usuarios finales, de usuarios finales individuales o de aplicaciones.

7. Método de acuerdo con la reivindicación 1, en el que puede producirse una variación dinámica del umbral a medida que cambian las cargas de célula o de red.

8. Método de acuerdo con la reivindicación 1, en el que la variación dinámica del umbral puede producirse para motivar la adopción de un nuevo servicio de replicación de datos.

9. Método de acuerdo con la reivindicación 1, en el que el umbral puede variar en función de uno o varios de los siguientes elementos:

(a) la hora en curso
(b) el estado de itinerancia del dispositivo
(c) la carga de célula o de red
(d) el tiempo transcurrido desde la última replicación
(e) la tarifa del usuario.

10. Método de acuerdo con la reivindicación 1 en el que, si el peso de ningún objeto sobrepasa el umbral en un momento determinado, se calcula el intervalo de tiempo que transcurrirá antes de que el peso de un objeto sobrepase el umbral y se configura un temporizador para ese intervalo de tiempo.

11. Método de acuerdo con la reivindicación 10, en el que el intervalo de tiempo se vuelve a calcular si:

(a) se añade un nuevo objeto al registro de cambios

(b) se implementa un nuevo umbral
(c) expira el temporizador
(d) cambia una carga de célula o de red
(e) la memoria del dispositivo desciende por debajo de un nivel predefinido
(f) el dispositivo detecta que su estado de itinerancia cambia
(g) se activa una nueva aplicación en el dispositivo
(h) se interrumpe una conexión de red.

12. Método de acuerdo con la reivindicación 1, en el que el usuario final del dispositivo puede invalidar la temporización de replicación por defecto con respecto a un objeto específico o un tipo de objeto específico.

13. Método de acuerdo con la reivindicación 1, en el que a un objeto que se ha de replicar se le asigna un límite temporal hasta el cual debe producirse la replicación.

14. Método de acuerdo con la reivindicación 13, en el que el límite temporal es dinámico.

15. Método de acuerdo con la reivindicación 13, en el que el límite temporal se modifica si la memoria del dispositivo cambia o si el dispositivo se mueve mediante itinerancia a una nueva red.

16. Método de acuerdo con la reivindicación 1, en el que a un objeto que se ha de replicar se le asigna un periodo de validez que define un tiempo o duración después del cual o de la cual el objeto será automáticamente borrado si no se ha replicado.

17. Método de acuerdo con la reivindicación 1, en el que distintos parámetros posibilitan que el operador de red ofrezca a los usuarios finales distintos niveles de servicio de replicación de datos, cada uno asociado a una tarifa distinta.

18. Método de acuerdo con la reivindicación 1 en el que, una vez se ha replicado un objeto de iniciación de conexión, también se envían otros objetos a continuación en un registro de cambios y que están pendientes de replicación.

19. Método de acuerdo con la reivindicación 18, en el que una función de umbral de oportunismo determina los objetos adicionales que se envían.

20. Método de acuerdo con la reivindicación 19, en el que el umbral de oportunismo cambia si el dispositivo se encuentra en una red de itinerancia.

21. Método de acuerdo con la reivindicación 19, en el que el umbral de oportunismo cambia en función de si se sobrepasa un umbral de carga de célula de la célula en la que se encuentra el dispositivo.

22. Método de acuerdo con la reivindicación 19, en el que el umbral de oportunismo es aplicado de manera coherente por parte del dispositivo y el servidor, comunicándose cambios en el umbral a través de la red.

23. Método de acuerdo con la reivindicación 19, en el que el operador de red puede cambiar el umbral de oportunismo.

24. Método de acuerdo con la reivindicación 1, en el que el momento real de la replicación depende del estado del dispositivo móvil, el estado de la red y los parámetros.

25. Dispositivo móvil programado con *software* que adapta el dispositivo móvil para llevar a cabo las siguientes etapas cuando se ejecuta el *software*:

(i) listar en un registro de cambios todos los objetos en el dispositivo móvil que se han de replicar;
(ii) definir parámetros mediante la utilización de (a) un peso asociado a cada objeto que define la urgencia con la que ha de replicarse dicho objeto y (b) un umbral que depende del tiempo, en el que el umbral varía con el tiempo de tal forma que se hace un uso eficiente del ancho de banda disponible;
(iii) comparar localmente el peso de cada objeto con el umbral en un momento determinado, determinando el resultado de la comparación si se envía el objeto para la replicación o no en ese momento:
donde los criterios que son pertinentes en lo que se refiere a la urgencia con la que ha de replicarse un objeto están representados por dicho peso asociado a ese objeto.

26. Servidor programado con *software* que adapta el servidor para llevar a cabo las siguientes etapas cuando se ejecuta el *software*:

(i) listar en un registro de cambios todos los objetos en el servidor que se han de replicar;
(ii) definir los parámetros mediante la utilización de (a) un peso asociado a cada objeto que define la urgencia con la que ha de replicarse dicho objeto y (b) un umbral que depende del tiempo, en el que el umbral varía con el tiempo de tal forma que se hace un uso eficiente del ancho de banda disponible;
(iii) comparar localmente el peso de cada objeto con el umbral en un momento determinado, determinando el resultado de la comparación si se envía el objeto para la replicación o no en ese momento:
donde los criterios que son pertinentes en lo que se refiere a la urgencia con la que ha de replicarse un objeto están representados por dicho peso asociado a ese objeto.

27. *Software* informático que, cuando se ejecuta en un dispositivo móvil o en un servidor, adapta dicho dispositivo móvil o servidor para llevar a cabo las siguientes etapas:

(i) listar en un registro de cambios todos los objetos en el dispositivo móvil o el servidor que se han de replicar;
(ii) definir los parámetros mediante la utilización de (a) un peso asociado a cada objeto que define la urgencia con la que ha de replicarse dicho objeto y (b) un umbral que depende del tiempo, en el que el umbral varía con el tiempo de tal forma que se hace un uso eficiente del ancho de banda disponible;
(iii) comparar localmente el peso de cada objeto con el umbral en un momento determinado, determinando el resultado de la comparación si se envía el objeto para la replicación o no en ese momento:
donde los criterios que son pertinentes en lo que se refiere a la urgencia con la que ha de replicarse un objeto están representados por dicho peso asociado a ese objeto.

Figura 1

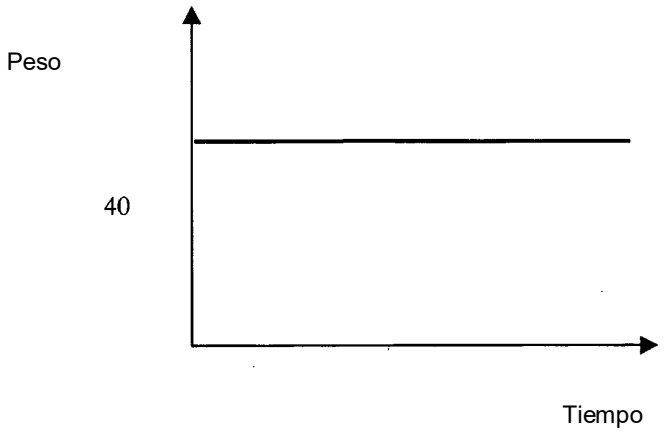

Figura 2

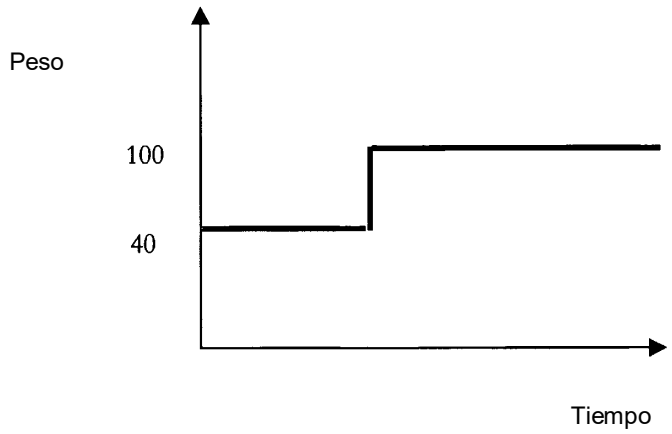

Figura 3