(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 747 962**

(51) Int. CI.:

G06F 12/02 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **29.06.2015 E 15174377 (0)**

(97) Fecha y número de publicación de la concesión europea: **24.07.2019 EP 3113026**

(54) Título: **Gestión de memoria automática que usa una unidad de gestión de memoria**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
12.03.2020

(73) Titular/es:

AICAS GMBH (100.0%)
Emmy-Noether-Straße 9
76131 Karlsruhe, DE

(72) Inventor/es:

SIEBERT, FRIDTJOF

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Gestión de memoria automática que usa una unidad de gestión de memoria

Campo técnico

La descripción hace referencia a los ordenadores, en general, y a los ordenadores con gestión automática de memoria (AMM - Automatic Memory Management, en inglés), en particular. La descripción también hace referencia a un método, a un producto de programa informático y a un sistema informático.

Antecedentes

Los sistemas informáticos ejecutan aplicaciones que tienen objetos, tales como datos (por ejemplo, textos, imágenes, mapas de bits, señales de control, campos de datos, conjuntos de datos, tablas) y código de programa (por ejemplo, para llamadas de rutina, tareas, subprocesos). El sistema informático almacena los objetos en la memoria de acceso aleatorio (RAM - Random Access Memory, en inglés) y utiliza direcciones lógicas y físicas para identificar los objetos dentro de la memoria. Los objetos generalmente hacen referencia unos a otros. Las relaciones de referencia entre los objetos se pueden describir mediante un llamado gráfico de objeto.

Puesto que las aplicaciones se ejecutan de manera continua, agregan objetos a la memoria, modifican objetos en la memoria, eliminan objetos de la memoria y/o cambian las relaciones de referencia entre los objetos. El almacenamiento de objetos no es estático, sino bastante dinámico.

En consecuencia, el espacio de memoria se asigna a algunos objetos y se desasigna de otros objetos siempre que se ejecute la aplicación. Con el tiempo, la asignación / desasignación conduce a una fragmentación de la RAM, y la memoria libre ya no ocupa un rango contiguo de direcciones. La fragmentación impide la asignación contigua de memoria a objetos grandes. Dividir los objetos en porciones no es una opción, debido a una variedad de otras razones. Las técnicas de fragmentación incluyen las llamadas etapas de compactación, también denominadas compactación, en lo que sigue, para mover objetos dentro de la memoria o dentro de espacios de direcciones.

Es normal en la técnica llevar a cabo asignaciones / desasignaciones y/o desfragmentación mediante módulos de gestión automática de memoria (AMM). Los módulos de AMM se ejecutan en paralelo a las aplicaciones o están entrelazados con las aplicaciones. Los módulos de AMM funcionan de acuerdo con reglas predefinidas. AMM también se conoce en la técnica como "recolección de basura". Los módulos de AMM u otros programas o herramientas que proporcionan AMM también se conocen como "recolectores de basura".

Tal como se mencionó, las aplicaciones cambian las relaciones de referencia entre los objetos. En términos de AMM, la aplicación muta el gráfico del objeto y, por lo tanto, las aplicaciones se denominan "aplicaciones de mutación".

En un primer escenario, las aplicaciones (es decir, las aplicaciones de mutación) se ejecutan directamente en los ordenadores, y los módulos de AMM se implementan como funciones de los sistemas operativos de los ordenadores.

En un segundo escenario, las aplicaciones necesitan un entorno de tiempo de ejecución (RTE - Run-time Environment, en inglés) para su ejecución. El módulo de AMM se puede implementar como una función del RTE. En ese caso, el módulo de AMM se ejecuta como un proceso fuera de los sistemas operativos (de los ordenadores). Por ejemplo, el RTE puede estar especializado para un lenguaje de programación particular, tal como JAVA.

No obstante, en ambos escenarios, el módulo de AMM consume recursos del ordenador, tales como tiempo de procesador y memoria. Hay un conflicto de recursos que tiene al menos dos aspectos: mientras el módulo de AMM lleva a cabo asignación, desasignación, desfragmentación o similares, la aplicación debe ser detenida o pausada. Como alternativa, el módulo de AMM debe ser programado con más complejidad, para permitir su ejecución en paralelo.

De manera más detallada, mover un objeto (durante la compactación mencionada anteriormente) impide que la aplicación acceda al objeto mientras se está moviendo el objeto. Esto provoca pausas en la ejecución continua de la aplicación. Además, el tiempo que lleva mover un objeto dentro de la RAM depende del tamaño del objeto. Esta dependencia del tiempo en función del tamaño suele ser una relación lineal: mover un objeto grande lleva más tiempo que mover un objeto pequeño. Permitir que se muevan objetos de un tamaño arbitrario puede hacer que las aplicaciones se detengan durante intervalos de tiempo arbitrarios.

Además, de nuevo en ambos escenarios, la AMM podría provocar el mal funcionamiento de una aplicación por varias razones, tales como, por ejemplo: (i) La fragmentación puede evitar la asignación de espacio de memoria contiguo a los objetos. (ii) El módulo de AMM puede provocar una pausa en la ejecución de la aplicación o las aplicaciones. (iii) El módulo de AMM puede fallar al identificar la memoria para su reutilización (reclamar memoria no utilizada) con la suficiente rapidez.

Algunas aplicaciones se utilizan en telecomunicaciones, en el control de máquinas industriales o en otras áreas,

para que la disponibilidad continua de la aplicación en tiempo real sea obligatoria. Por ejemplo, algunas aplicaciones requieren la ejecución periódica de tareas (períodos tales como de 100 microsegundos).

De acuerdo con el documento US 2012/102289 A1, los objetos de diferentes tamaños, tipos u otros aspectos pueden tratarse de manera diferente. Por ejemplo, los objetos pequeños y grandes se distinguen y la memoria es asignada en los extremos opuestos de la memoria de almacenamiento dinámico. Este enfoque reduce el trabajo necesario para la compactación. No obstante, esto se lleva a cabo sin utilizar una unidad de gestión de memoria (MMU - Memory Management Unit, en inglés).

Según el documento US 2010/250893 A1, la utilización de una MMU evita la copia de la memoria física. Este enfoque permite compactar direcciones virtuales de objetos sin copiar la memoria física subyacente. En cambio, nuevas direcciones virtuales de objetos son asignadas a las direcciones físicas originales. Esto evita la necesidad de copiar los datos físicos.

Compendio

Las realizaciones de la presente invención proporcionan un enfoque para resolver el conflicto mencionado anteriormente en un sistema informático con un módulo mejorado de gestión automática de memoria (AMM), un método implementado por ordenador y un producto de programa informático.

El enfoque aprovecha los desarrollos en la informática moderna: los sistemas informáticos tienen unidades de gestión de memoria (MMU), es decir, hardware que proporciona una asignación de direcciones de lógicas a físicas, y los sistemas informáticos tienen espacios de direcciones (por ejemplo, 264 direcciones) que son mayores que el tamaño de la memoria física real. El espacio (o "rango") para las direcciones lógicas está dividido en al menos dos subespacios: en un primer subespacio para esa compactación (durante la desfragmentación), el llamado "espacio de compactación", y en un segundo subespacio para esa compactación deshabilitada, el llamado "espacio de no compactación". En otras palabras, el primer subespacio es "compactable" y el segundo subespacio es "no compactable". El segundo subespacio es lo suficientemente grande como para alojar todos los objetos potenciales de todas las categorías de tamaño, y la MMU asigna el segundo subespacio para desasignar la memoria física sin la necesidad de mover objetos dentro de la RAM.

Cuando la aplicación / aplicación de mutación solicita memoria para objetos particulares, el módulo de AMM aplica, al menos, un criterio de distinción predefinido y distingue los objetos en "objetos de compactación" y "objetos sin compactación", y asigna direcciones lógicas en el espacio de compactación, o en el espacio de no compactación, respectivamente.

El criterio de distinción está relacionado con el conflicto mencionado anteriormente: para abordar el conflicto de recursos, los objetos se distinguen por el tamaño del objeto. Hay objetos que tienen un tamaño inferior al umbral y objetos que tienen un tamaño superior (o igual) al umbral. En la descripción, los objetos se distinguen como "objetos pequeños" y "objetos grandes", respectivamente.

El módulo de AMM asigna direcciones lógicas en el espacio de compactación a los objetos pequeños, y asigna direcciones lógicas en el espacio de no compactación a los objetos grandes.

La dependencia de tiempo con respecto al tamaño mencionada anteriormente se considera en el sentido de que se puede evitar la compactación de una porción de la RAM que almacena objetos grandes, o al menos puede estar limitada a acciones de la MMU (es decir, hardware) que no compiten por los recursos con el módulo de AMM. El tamaño del umbral puede ser el límite de compactación que diferencia la asignación de direcciones lógicas en el espacio de compactación y no compactación.

Opcionalmente, los objetos se distinguen de acuerdo con una combinación de criterios.

El módulo de AMM aplica compactación a los objetos pequeños en direcciones lógicas en el espacio de compactación, pero no necesita aplicar compactación a los objetos grandes (en direcciones lógicas) en el espacio de no compactación. En vista del conflicto de recursos, se observa que el tiempo requerido por el módulo de AMM (en combinación con la MMU y la RAM) para la compactación es limitado. Este tiempo está sustancialmente relacionado linealmente con el tamaño del objeto más grande (que tiene su dirección lógica en el espacio de compactación). Por lo tanto, el tiempo durante el cual la aplicación tendría que detenerse puede ser limitado.

Múltiples aplicaciones de mutación pueden acceder al mismo objeto (espacio sin compactación) en direcciones que permanecen sin cambios en el tiempo extra. No es necesario propagar actualizaciones de direcciones a las múltiples aplicaciones de mutación y entre las mismas.

Los parámetros de implementación pueden tener en cuenta los detalles del sistema informático y/o la aplicación: en caso de distinción de objetos por tamaño de objeto, el espacio de compactación tiene direcciones lógicas que están separadas de acuerdo con el tamaño del umbral. El tamaño del umbral puede estar relacionado con una granularidad de página de la RAM.

Se pueden implementar realizaciones alternativas. Por ejemplo, en lugar de habilitar / deshabilitar la compactación, o la compactación / no compactación, la compactación se puede llevar a cabo mediante diferentes modos de compactación, tales como la compactación en intervalos relativamente cortos (primer subespacio, solamente) y la compactación en intervalos relativamente grandes (ambos subespacios).

Un método implementado por ordenador para operar un módulo de gestión automática de memoria (AMM) en un sistema informático comprende recibir, asignar, asignar (map, en inglés) y compactar. En la etapa de recepción, el módulo de AMM recibe solicitudes de asignación de memoria de una aplicación de mutación. Las solicitudes están previstas para que objetos particulares se almacenen en una memoria de acceso aleatorio del sistema informático. En la etapa de asignación, el módulo de AMM asigna direcciones lógicas particulares dentro de un espacio de direcciones lógicas a los objetos particulares. El módulo de AMM distingue, de este modo, los objetos particulares de acuerdo con al menos un criterio, y asigna direcciones lógicas de un primer subespacio y direcciones lógicas de un segundo subespacio. En una etapa de asignación, una unidad de gestión de memoria (MMU) asigna las direcciones lógicas asignadas desde el segundo subespacio a la memoria física en la memoria de acceso aleatorio. En la etapa de compactación, la MMU compacta direcciones lógicas dentro del primer subespacio a la vez que mueve los objetos correspondientes dentro de la memoria de acceso aleatorio.

En la asignación, el módulo de AMM distingue, opcionalmente, los objetos particulares de acuerdo con que el al menos un criterio sea el tamaño del objeto.

En la asignación, el módulo de AMM utiliza, opcionalmente, un tamaño umbral predefinido como criterio de tamaño de objeto, y asigna direcciones lógicas en el primer subespacio a objetos que tienen un tamaño de objeto por debajo del tamaño umbral predefinido, y asigna direcciones lógicas en el segundo subespacio a objetos que tienen un tamaño de objeto por encima del tamaño umbral predefinido o igual al tamaño umbral predefinido.

Opcionalmente, el tamaño umbral predefinido puede estar relacionado con un tamaño de página de la memoria de acceso aleatorio. El tamaño umbral predefinido puede ser igual al tamaño de página o a un múltiplo del tamaño de página.

En una etapa adicional de reserva, el módulo de AMM puede reservar el segundo subespacio del espacio de direcciones lógicas, con porciones que tienen ranuras para objetos de diferentes clases de tamaño. La reserva se puede llevar a cabo para porciones del mismo tamaño. La reserva puede incluir el cálculo del tamaño de una porción en correspondencia con un tamaño de almacenamiento dinámico máximo de la aplicación de mutación, que es el tamaño acumulado de los objetos que potencialmente se almacenarán en el segundo subespacio. La reserva puede incluir reservar porciones. Una primera porción puede tener un primer número de ranuras que corresponde al tamaño del montón dividido por el umbral predeterminado, y una segunda porción puede tener un segundo número de ranuras que corresponde al primer número dividido por 2. De manera alternativa, la reserva puede incluir reservar porciones, teniendo una porción de inicio un primer número de ranuras como un número de potencia de 2, y ranuras adyacentes que tienen números de ranuras calculadas en una serie geométrica a partir del primer número de ranuras.

El método puede tener las etapas adicionales para desasignar algunas de las direcciones lógicas particulares desde cualquier subespacio. El módulo de AMM ordena a la MMU que elimine la correspondencia de la memoria física de las direcciones lógicas asignadas previamente en el segundo subespacio sin compactar los objetos en la memoria de acceso aleatorio.

Opcionalmente, el módulo de AMM puede distinguir los objetos particulares de acuerdo con que al menos un criterio sea un atributo de objeto. El atributo de objeto puede comprender atributos seleccionados entre los siguientes: tipo de objeto, frecuencia de acceso a objetos, el objeto que se pasa a código externo y el hardware al que accede directamente el objeto.

Un sistema informático puede llevar a cabo las etapas del método implementado por ordenador, y un producto de programa informático que, cuando es cargado en la memoria de un ordenador y ejecutado al menos un procesador del ordenador, lleva a cabo las etapas del método implementado por ordenador. El producto de programa informático puede ser proporcionado de tal manera que el módulo de AMM forme parte de un entorno de tiempo de ejecución para aplicaciones de mutación escritos en un lenguaje de programación tal como: JAVA, C#, PYTHON, JAVASCRIPT, SCALA o GO.

Breve descripción de los dibujos

La figura 1 ilustra un diagrama de bloques de un sistema informático con un procesador, una memoria de acceso aleatorio y una unidad de gestión de memoria, así como un módulo de aplicación de mutación y un módulo de gestión automática de memoria (AMM);

la figura 2 ilustra un espacio de direcciones lógicas del módulo de AMM en el sistema informático en momentos consecutivos;

la figura 3 ilustra el diagrama de bloques del sistema informático, pero con dos aplicaciones de mutación y un

módulo de AMM que tiene dos espacios de direcciones lógicas;

la figura 4 ilustra un diagrama de flujo de un método implementado por un ordenador;

la figura 5 ilustra un subespacio de direcciones lógicas que está dividido en múltiples porciones; y

la figura 6 ilustra un ejemplo de un dispositivo informático genérico y un dispositivo informático móvil genérico, que puede ser utilizado con las técnicas descritas en el presente documento.

Descripción detallada

La descripción detallada comienza a explicar el diagrama general. La descripción se centra en resolver el conflicto de recursos, siendo el criterio de distinción el tamaño del objeto (SO - Object Size, en inglés), e ilustra el funcionamiento del módulo de AMM a lo largo del tiempo, a modo de ejemplo. La descripción explica, además, un diagrama de flujo, así como los parámetros de implementación, y se cierra describiendo un sistema informático general.

La figura 1 ilustra un diagrama de bloques del sistema informático 100 con el procesador 110 (tal como una unidad central de procesamiento (CPU - Central Processing Unit, en inglés)), una memoria de acceso aleatorio (RAM) 120 y una unidad de gestión de memoria (MMU) 130. Para simplificar, no se ilustran otros componentes de hardware, tales como buses para comunicar datos y/o direcciones. El sistema informático 100 es un sistema que puede ser utilizado para implementar el método implementado por ordenador (véase la figura 4).

La figura 1 también ilustra módulos de software (mediante rectángulos con líneas discontinuas): la aplicación de mutación 140 y el módulo de AMM 150. Los módulos de software son implementados en la RAM 120 y ejecutados mediante el procesador 110, y son posibles otras implementaciones. Para simplificar, la figura 1 ilustra los módulos fuera de la RAM 120. El módulo de software y el hardware interactúan con el sistema operativo del sistema informático, por ejemplo, para implementar realmente las solicitudes. Las personas expertas en la técnica pueden considerar el sistema operativo sin la necesidad de una explicación adicional en este documento.

La aplicación de mutación 140 es implementado mediante código de programa, es decir, código para una aplicación. La aplicación de mutación 140 puede ser una aplicación que necesita un entorno de tiempo de ejecución (RTE, en ocasiones: máquina virtual) para su ejecución. Por ejemplo, la aplicación de mutación 140 puede estar escrita en un lenguaje de programación tal como JAVA, C#, PYTHON, JAVASCRIPT, SCALA o GO. El módulo de AMM 150 puede formar parte de ese RTE.

Para simplificar, se supone que el procesador 110 puede ejecutar solo un módulo de software en un momento determinado. Esto lleva al requisito de limitar el tiempo durante el cual el módulo de AMM 150 (y los componentes que interactúan con el módulo de AMM, tales como MMU y RAM) están en funcionamiento exclusivamente (ya sea el módulo de AMM, solamente, no la aplicación de mutación; o viceversa). Esta limitación en el tiempo se simboliza en el presente documento como tiempo máximo permitido (o tiempo acotado) T_{MAX}, que, en general, se mide en microsegundos. La limitación puede ser relativa: T_{MAX} puede ser una fracción del tiempo de la aplicación de mutación (con respecto al tiempo de ejecución de la aplicación de mutación). La limitación también puede ser un intervalo de tiempo absoluto, para garantizar la disponibilidad del procesador para la aplicación de mutación durante períodos predefinidos.

La aplicación de mutación 140, que está siendo ejecutada por el procesador 110, emite solicitudes de asignación de memoria 141 ("Req [i]") para que los objetos se almacenen temporalmente en la RAM 120. Los objetos pueden ser objetos de datos (por ejemplo, datos de imagen, datos de audio, datos de video, datos de control) y/u objetos de código (por ejemplo, código para rutinas de programa), o cualquier otra colección de bits y bytes que serán almacenados en la RAM 120. A continuación, los objetos se etiquetan como 0[1], 0[2], ... 0[i], ... de acuerdo con el orden en que la aplicación de mutación 140 emite las solicitudes.

El módulo de AMM 150 comprende un espacio de direcciones lógicas 155 con direcciones lógicas Ax.y. Tal como se utiliza en el presente documento, la notación "Ax.y" se simplifica y puede incluir una dirección de inicio y una dirección de finalización, una dirección de inicio y un rango que permite el cálculo de la dirección de finalización, un espacio en un espacio de direcciones o cualquier otra convención, que identifican direcciones lógicas.

En el ejemplo, la distancia, medida en bytes, entre las direcciones consecutivas Ax.y y Ax.y + 1 es el "espacio entre direcciones", dado como delta x (por ejemplo, delta 1, delta 2).

El espacio de direcciones lógicas 155 es asignado a la aplicación de mutación 140. El módulo de AMM 150 comprende, además, el módulo de función 156, para llevar a cabo, al menos, las funciones de ASIGNACIÓN, DESASIGNACIÓN y COMPACTACIÓN SELECTIVA. El estado del espacio de direcciones lógicas 155 a lo largo del tiempo en vista de estas funciones se ilustra mediante un ejemplo en la figura 2.

En la función ASIGNACIÓN, el módulo de AMM 150 recibe solicitudes de asignación de memoria 141 de la aplicación de mutación 140 y asigna direcciones lógicas Ax.y dentro del espacio de direcciones lógicas 155 a los

objetos. La asignación de direcciones lógicas Ax.y es selectiva, al menos, en un primer subespacio 155-1 (direcciones lógicas A1.1 ... A1.K) y un segundo subespacio 155-2 (direcciones lógicas A2.1 ... A2.L). La distinción de los subespacios está relacionada con la función COMPACTACIÓN (en un momento posterior) de tal manera que el primer subespacio 155-1 también se denomina "espacio de compactación" y el segundo subespacio 155-2 se denomina "espacio de no compactación". La distinción depende, al menos, de un criterio. Tal como se mencionó anteriormente, la figura 1 está centrada en el criterio del tamaño del objeto.

La MMU 130 tiene una función de asignación (mapeo, en inglés) para asignar direcciones lógicas Ax.y del espacio de direcciones 155 a direcciones físicas (P1, P2, ...) de la RAM 120. La MMU 130 gestiona páginas de un tamaño fijo. Los tamaños de página habituales pueden ser, por ejemplo, 4KB (Kilobyte) o 128 MB (Megabyte). El ejemplo de la figura 1 supone un tamaño de página de 4KB. Tal como se mencionó anteriormente, las direcciones lógicas en el espacio de no compactación pueden ser asignadas a objetos grandes (es decir, objetos más grandes que el tamaño de página). La MMU 130 asigna algunas o más páginas de memoria a direcciones lógicas contiguas. En el ejemplo, el objeto 0[1] tiene un tamaño de objeto de 8 KB (o 2 páginas), y el objeto 0[2] también tiene 8 KB. 0[1] es un objeto en el espacio de no compactación y requiere 2 páginas de memoria. Tal como se ilustra con flechas de puntos, las direcciones lógicas A2.1a y A2.1b se asignan al objeto 0[1]; y la MMU 130 asigna A2.1a a P1 (que tiene un tamaño de una página), y asigna A2.1b a P2 (que también tiene un tamaño de una página). Además, las direcciones lógicas A2.2a y A2.2b son asignadas al objeto 0[2]; y la MMU 130 asigna A2.2a a P6 (que tiene un tamaño de una página), y asigna A2.2b a P10 (que también tiene un tamaño de una página). Los objetos que tienen un tamaño de objeto diferente (por ejemplo, N páginas por objeto) pueden ser asignados en consecuencia.

La dirección lógica A1.1 (del primer subespacio) es asignada a la dirección física P101. Para simplificar, no se ilustran más asignaciones.

El mapeo de lógico a físico (en la MMU 130) para direcciones lógicas en el subespacio 155-1 (compactación) puede ser estático, no necesita cambiar para satisfacer las solicitudes de asignación.

La figura 1 ilustra que los objetos se almacenan en la memoria dentro de la RAM 120 y también ilustra que la RAM 120 puede ser fragmentada. Por ejemplo, las direcciones físicas P1 y P2 son contiguas (sin otras direcciones intermedias), y las direcciones físicas P6 y P10 no son contiguas (están separadas por P7, P8, P9). La MMU 130 puede alojar una fragmentación para objetos grandes, en la RAM. Pero debido a la asignación, un solo objeto puede tener una única dirección lógica, de tal manera que la aplicación de mutación 140 puede acceder a dicho único objeto en su única dirección lógica (por ejemplo, 0[2] en A2.2). Para el rendimiento de la aplicación de mutación 140, puede ser una ventaja tener direcciones lógicas contiguas. En el ejemplo, las direcciones lógicas A2.1 y A2.2 son contiguas, pero la no contigüidad de los objetos que se almacenan físicamente en la RAM está alojada mediante la MMU 130.

Mirando la figura de izquierda a derecha, la aplicación de mutación 140 emite solicitudes de asignación de memoria 141 (Req [1], Req [2] ...) para objetos particulares 0[1], 0[2], y así sucesivamente. El módulo de AMM 150 recibe solicitudes 141 y asigna una dirección lógica Ax.y a cada objeto. La MMU 130 proporciona la asignación a direcciones físicas, de tal manera que los objetos se almacenan en la RAM 120.

La figura 1 ilustra un momento en el que se han asignado direcciones lógicas a los objetos 0[1] a 0[9], y en que estos objetos se han almacenado en las direcciones físicas correspondientes en la RAM 120.

La aplicación de mutación 140 lee, crea, elimina objetos o cambia referencias entre diferentes objetos 0[i] (no ilustrado) y 0[i] con frecuencia. Como su nombre indica, la aplicación de mutación 140 muta continuamente el gráfico del objeto, es decir, la aplicación de mutación 140 cambia las referencias entre los objetos. Como consecuencia, algunos objetos pueden volverse inalcanzables desde el gráfico de objetos y, por lo tanto, ser inaccesibles por la aplicación de mutación 140, por lo que ya no necesitan permanecer en la RAM (los llamados "objetos basura"), mientras que los objetos alcanzables o accesibles deben permanecer en la RAM (los llamados "objetos vivos").

Ejecutando la función de DESASIGNACIÓN, el módulo de AMM 150 desasigna periódicamente direcciones lógicas de objetos que han sido identificados como objetos basura por el módulo de AMM 150. Puesto que la asignación y la desasignación a lo largo del tiempo conducen a la fragmentación, ejecutando la función de COMPACTACIÓN, el módulo de AMM 150 compacta el espacio de direcciones lógicas reasignando objetos (vivos) a otras direcciones lógicas Ax.y. Para alojar el conflicto mencionado anteriormente, el módulo de AMM 150 lleva a cabo la función de COMPACTACIÓN solo para el primer subespacio. La MMU 130, por lo tanto, se utiliza, opcionalmente. En lugar de compactar, el módulo de AMM 150 puede combatir la fragmentación (primer subespacio) de lo contrario, mediante técnicas conocidas en la técnica, por ejemplo, mediante la recolección tradicional de basura.

La MMU 150 puede diferenciar la solicitud de asignación de memoria Req [i] de acuerdo con el tamaño del objeto 0[i] (siendo el criterio). El tamaño de un objeto se puede medir en unidades de medida tales como Byte (B), Kilobyte (KB), Megabyte (MB), Gigabyte (GB), etc. Un objeto con un tamaño inferior al tamaño umbral (TS - Threshold Size, en inglés) es asignado a las direcciones lógicas A1.y ("espacio de compactación"). Durante la compactación, este pequeño objeto (si es un objeto vivo) tendría que moverse dentro de la RAM 120, pero el tiempo que lleva moverlo está por debajo del tiempo máximo permitido T_{MAX}. Un objeto con un tamaño superior (o igual) al tamaño umbral

(TS) es asignado a las direcciones lógicas A2.y ("no compactación"), de tal manera que el objeto grande no se movería dentro de la RAM 120. En caso de que el objeto se identifique como basura, el módulo de AMM 150 simplemente liberaría la memoria física eliminando la asignación de direcciones lógicas (de los objetos) a la memoria física dentro de la MMU 130.

Las direcciones lógicas A1.y están separadas, al menos, por delta 1, en el ejemplo de la figura 1, delta 1 corresponde al tamaño umbral TS (delta 1 = TS). El tamaño umbral TS puede ser igual a la granularidad de una página en la RAM 120. Por ejemplo, TS puede ser igual al tamaño de página PS = 4 KB.

Las direcciones lógicas A2.y están separadas, al menos, por delta 2. Para las direcciones lógicas A2.y, se puede implementar delta 2 para diferentes clases de tamaño. La figura 1 ilustra delta 2' para A2.L. Los detalles para tal enfoque que diferencia aún más los objetos grandes se describen con respecto a la figura 5)

La figura 2 ilustra el espacio de direcciones lógicas 155 (subespacios 155-1, 152-2) del módulo de AMM 150 en el sistema informático 100 de la figura 1 en momentos consecutivos. Se supone que el módulo de AMM 150 se diferencia de acuerdo con el criterio del tamaño del objeto y distingue los objetos pequeños (< TS) de los objetos grandes (> = TS).

En el momento t[2], el módulo de AMM 150 ha recibido Req [1] y Req [2] y ha asignado A2.1 al objeto 0[1], y A2.2 al objeto 0[2]. Ambos objetos tienen un tamaño superior a TS, pero un tamaño que se ajusta a delta 2.

En el momento t [4], el módulo de AMM 150 ha recibido, además, Req [3] y Req [4], y ha asignado A2.3 al objeto 0[3] y A2.4 al objeto 0[4]. De nuevo, ambos objetos tienen un tamaño superior a TS, pero ajustándose a delta2.

En el momento t[10], el módulo de AMM 150 ha recibido, además, Req [5] a Req [10], y ha asignado la dirección lógica A2.5 a 0[8] (objeto grande); y A1.1 a 0[5], A1.2 a 0[6], A1.3 a 0[7], A1.4 a 0[9] y A1.5 a 0[10] (objetos pequeños). Las direcciones son contiguas: A2.1 a A2.5 (es decir, sin espacio) para 0[1], 0[2], 0[3], 0[4], 0[8] y A1.1 a A1.4 (es decir, sin espacio) para 0[5], 0[6], 0[7], 0[9], 0[10].

En un momento posterior t', el módulo de AMM 150 ha determinado que los objetos 0[3], 0[4], 0[6] y 0[9] son objetos basura. Como consecuencia, el módulo de AMM 150 ha desasignado las direcciones lógicas A2.3, A2.4, ha ordenado a la MMU 130 que elimine la correspondencia correspondiente a las direcciones físicas. El módulo de AMM 150 también ha desasignado A1.2, A1.4, pero sin interacción con la MMU 130 o la AM 120). Debido a la desasignación, los espacios de direcciones en ambos subespacios 155-1 y 155-2 ya no son contiguos.

En un momento posterior t", el módulo de AMM 150 ha compactado las direcciones en el primer espacio 155-1 (solo) pero no ha compactado las direcciones en el segundo espacio 155-2. La compactación incluye la interacción con la RAM 120 para que los objetos se muevan. No obstante, el tiempo que lleva mover uno de los objetos pequeños 0[5], 0[7] u 0[10] es menor que T_{MAX}.

Después de esa breve interrupción de tiempo, la aplicación de mutación 140 continúa ejecutándose y envía más solicitudes. El módulo de AMM 150 puede continuar asignando direcciones para otros objetos, tales como objetos grandes (por ejemplo, 0[14], 0[15]).

Aunque las figuras 1 a 2 se centran en el tamaño del objeto como criterio, la AMM 150 también puede distinguir objetos particulares de acuerdo con un atributo de objeto (OA - Object Attribute, en inglés). El atributo de objeto (OA) puede comprender atributos tales como (1) tipo de objeto, (2) frecuencia de accesos al objeto, (3) una distinción entre los objetos que son pasados al código que es externo a la aplicación de mutación y los objetos que se procesan dentro de la aplicación de mutación solamente, y (4) el objeto al que se accede directamente mediante hardware.

La figura 3 ilustra el diagrama de bloques del sistema informático 100, pero con dos aplicaciones de mutación 140 y 140' y un módulo de AMM 150 que tiene dos espacios de direcciones lógicas 155 y 155'.

El módulo de AMM 150 asigna espacios de direcciones 155 y 155' que pueden tener subespacios (compactos, no compactos). Los tamaños generales de los espacios de direcciones pueden ser diferentes entre sí. El módulo de AMM 150 recibe solicitudes de asignación de memoria 141, 141' (Req) para las aplicaciones de mutación 140, 140'. El módulo de función 156 puede ser común para ambos espacios.

La figura 4 ilustra un diagrama de flujo del método 400 implementado por un ordenador para operar el módulo de AMM 150 en el sistema informático 100 (véase la figura 1).

En una etapa de recepción 420, el módulo de AMM 150 recibe solicitudes de asignación de memoria para objetos particulares de la aplicación de mutación. Los objetos particulares deben ser almacenados en la memoria de acceso aleatorio del sistema informático.

En una etapa de asignación 430, el módulo de AMM 150 asigna direcciones lógicas particulares dentro del espacio de direcciones lógicas a los objetos particulares. De este modo, el módulo de AMM 150 distingue 435 los objetos particulares de acuerdo con, al menos, un criterio (por ejemplo, umbral TS, atributo de objeto OA), y asigna 431, 432

direcciones lógicas (por ejemplo, A1.y) de un primer subespacio (espacio de compactación) y direcciones lógicas (por ejemplo, A2.y) de un segundo subespacio (espacio sin compactación).

En una etapa de asignación 442, el módulo de AMM 150 ordena a la unidad de gestión de memoria 130 que asigne las direcciones lógicas asignadas desde el segundo subespacio (es decir, el espacio sin compactación) a la memoria física en la RAM.

En una etapa de compactación 451, el módulo de AMM 150 compacta objetos vivos en direcciones lógicas dentro del primer subespacio (espacio de compactación) en combinación con mover objetos correspondientes en la memoria de acceso aleatorio. Por el contrario, el módulo de AMM 150 no compacta objetos vivos en direcciones lógicas dentro del segundo subespacio (espacio sin compactación), sino que, en este caso, la MMU reasigna memoria física no utilizada a direcciones lógicas que se utilizan para objetos recién asignados, sin mover objetos dentro de la RAM.

En la etapa de asignación 430, el módulo de AMM 150 puede distinguir 435 los objetos particulares de acuerdo con que el al menos un criterio sea un criterio de tamaño de objeto. Un tamaño umbral predefinido (TS) puede ser el criterio de tamaño de objeto, y el módulo de AMM 150 puede asignar 431 direcciones lógicas en el primer subespacio a objetos que tienen un tamaño de objeto por debajo del tamaño umbral (TS) predefinido y puede asignar 432 direcciones lógicas en el segundo subespacio a objetos que tienen un tamaño de objeto superior a TS o igual a TS. El tamaño umbral predefinido puede estar relacionado con el tamaño de página (PS - Page Size, en inglés) de la RAM 120, y puede ser igual a PS.

La figura 4 también ilustra el funcionamiento de la MMU 130 que lleva a cabo la asignación 441/442, lo que significa traducir direcciones lógicas a la memoria física. Ambas etapas de asignación se aplican una vez que el módulo de AMM 150 ha asignado una dirección lógica, pero con la distinción de asignación 441 utilizando direcciones lógicas del espacio de compactación, y asignación 442 utilizando direcciones lógicas del espacio de no compactación. La asignación 441 se considera estática, sin cambios a lo largo del tiempo; pero la asignación 442 es, en realidad, una reasignación continua que asigna de manera dinámica la memoria física a las direcciones lógicas actualmente en utilización por los objetos vivos. En la etapa de compactación 451, la MMU 130 mueve objetos vivos (que tienen sus direcciones lógicas asignadas en el espacio de compactación) dentro de la memoria. La MMU 140 mueve los objetos de tal manera que la memoria lógica libre (dentro del espacio de compactación) esté disponible en rangos contiguos que son lo suficientemente grandes como para utilizar asignaciones hasta el tamaño de página (PS).

Volviendo de nuevo al funcionamiento del módulo de AMM 150, en una etapa adicional que reserva 410, el módulo de AMM 150 reserva el segundo subespacio del espacio de direcciones lógicas con porciones que tienen ranuras para objetos de diferentes clases de tamaño (véase la figura 5). La reserva 410 puede ser llevada a cabo para porciones del mismo tamaño.

La reserva 410 puede incluir el cálculo del tamaño de una porción en correspondencia con un tamaño de almacenamiento dinámico máximo de la aplicación de mutación, que es el tamaño acumulado de los objetos que, potencialmente, se almacenarán en el segundo subespacio. La reserva 410 puede incluir reservar porciones, en las que una primera porción tiene un primer número de ranuras que corresponde al tamaño de almacenamiento dinámico (HS - Heap Size, en inglés) dividido por el umbral predeterminado (TS), y una segunda porción tiene un segundo número de ranuras que corresponde al primer número dividido por 2. En una alternativa, una porción de inicio tiene el primer número de ranuras que es un número de potencia de 2, y las ranuras adyacentes tienen números calculados en una serie geométrica a partir del primer número.

El método 400 puede tener etapas adicionales para desasignar algunas de las direcciones lógicas particulares de cualquier subespacio. El módulo de AMM 150 ordena a la MMU que elimine la correspondencia de la memoria física de las direcciones lógicas asignadas previamente en el segundo subespacio sin compactar los objetos en la RAM.

Opcionalmente, el módulo de AMM 150 puede distinguir 435 los objetos particulares de acuerdo con que al menos un criterio sea un atributo de objeto (OA), con atributos tales como prioridad de ejecución, frecuencia de ejecución, tipo de archivo, la utilización del objeto para pasar datos a código externo (escrito en otro idioma o accesible por hardware).

La figura 4 también ilustra un programa informático o un producto de programa informático. El producto de programa informático, cuando es cargado en la memoria de un ordenador y ejecutado por al menos un procesador del ordenador, lleva a cabo las etapas del método 400 implementado por el ordenador. En realizaciones, la memoria puede ser la RAM 120 y el procesador puede ser la CPU 110.

La descripción se convierte, a continuación, en aspectos de implementación que se centran en los tamaños de objeto que se consideran en la etapa de reserva 410.

Tal como se mencionó anteriormente, la figura 1 ilustra que los objetos en la RAM 150 no están fragmentados. No obstante, es posible mantener fragmentados los objetos grandes y, tal como se ha descrito anteriormente, llevar a cabo la compactación solo para los objetos pequeños.

La figura 5 ilustra un subespacio 555-2 de direcciones lógicas que está dividido en múltiples porciones. El espacio de direcciones lógicas 555-2 corresponde al espacio de direcciones lógicas 155-2 de la figura 1, es decir, al espacio de no compactación. En el ejemplo, las porciones tienen el mismo tamaño (medido en bytes), pero esta convención solo se utiliza para explicación.

Se utiliza la siguiente notación: A2.m.y representa una dirección lógica particular en una porción m particular, siendo y un identificador dentro de cada porción (véase la figura 1). Tal como se mencionó anteriormente, las direcciones lógicas A2.m.y pueden incluir direcciones de inicio y direcciones de finalización. En la figura 5, esto se ilustra para las porciones 7 y 8. Por ejemplo, la dirección de finalización de A2.7.2 tiene un desplazamiento de un byte por debajo de la dirección inicial de A2.8.1. Aunque se ilustra en columnas, se considera que las porciones forman un espacio de direcciones contiguo.

Las porciones tienen ranuras en diferentes clases de tamaño, medidas por el tamaño máximo de un objeto que se puede direccionar (tamaño máximo de objeto, MOS (Maximum Object Size, en inglés)).

La porción 8 tiene una dirección lógica A2.8.1 para un solo objeto, que tiene un tamaño máximo de objeto (MOS) de 512 KB. La porción 7 tiene 2 direcciones lógicas A2.2.1 y A2.2.2 para 2 objetos, cada uno con un MOS = 256 KB. La porción 6 tiene 4 direcciones lógicas A2.2.1 a A2.2.4 para 4 objetos, cada uno con un MOS = 128 KB. La porción 5 tiene 8 direcciones lógicas para MOS = 64 KB; la porción 4 tiene 16 direcciones lógicas para MOS = 32 KB, y así sucesivamente hasta la porción 1, que tiene 128 direcciones lógicas para MOS = 4 KB. Utilizar un esquema de potencia de 2 es conveniente, pero opcional.

El requisito de memoria general de la aplicación de mutación puede estar disponible como un tamaño máximo de almacenamiento dinámico (HS) que es el tamaño general de todos los objetos grandes (es decir, OS >= TS) al que, en cualquier momento dado durante la ejecución, la aplicación de mutación 140 continúa accediendo. El requisito de memoria para los objetos pequeños (es decir, OS < TS) se puede tener en cuenta por separado.

Puesto que el tamaño de cada porción del 1 al 8 corresponde al tamaño máximo de almacenamiento dinámico, la asignación de direcciones no fallará mientras la aplicación de mutación 140 permanezca dentro del tamaño máximo de almacenamiento dinámico (es decir, para emitir solicitudes de asignación de objetos con tamaños acumulados) de objetos vivos que exceden el tamaño de almacenamiento dinámico). En casos extremos, la aplicación de mutación 140 puede emitir una sola solicitud de asignación de memoria para un solo objeto (por ejemplo, 512 KB), o puede emitir solicitudes para 128 objetos con un tamaño de objeto de 4 KB. Generalmente, la aplicación de mutación 140 emite solicitudes de objetos que tienen diferentes tamaños.

El requisito general de espacio de direcciones lógicas se puede calcular, por ejemplo, como el número de porciones multiplicado por el tamaño (igual) de la porción, en el ejemplo como 8 * 512 KB. En comparación con el espacio de direcciones del ordenador (por ejemplo, 2^{64} tal como se mencionó anteriormente), este requisito general suele ser mucho menor.

En el ejemplo, la RAM 120 puede tener un esquema de direccionamiento habilitado para páginas con (por ejemplo), correspondiendo el tamaño de página (PS) al MOS de la porción 1, es decir, 4 KB.

El módulo de AMM 150 asigna el subespacio 555-2 con las porciones 1 a 8 al iniciarse. Por ejemplo, si el módulo de AMM 150 forma parte de un RTE (por ejemplo, Java Virtual Machine), el subespacio 555-2 es asignado al iniciar el RTE. También es posible asignar el subespacio 555-2 para cada nueva aplicación de mutación 140 (véase la figura 1, 140' en la figura 2) que se está lanzando. Tras el arranque, el módulo de AMM 150 generalmente aún no ha recibido solicitudes de asignación de memoria (Req) de la aplicación de mutación 140, y la MMU 130 aún no ha asignado las direcciones lógicas A2.m.y a la RAM 120.

La siguiente descripción hace referencia a las solicitudes de asignación de memoria (Req. Figura 1) para objetos grandes que se almacenarán en el segundo subespacio 155-2 (por ejemplo, el 555-2) solamente, y se supone que el módulo de AMM 150 utiliza el criterio de tamaño de objeto, con el umbral de 4 KB que corresponde a un tamaño de página PS = 4 KB de la RAM 120.

Tal como se describió anteriormente en detalles a la vista de la figura 1, tras la recepción de una solicitud de asignación de memoria (Req) para un objeto en particular, el módulo de AMM 150 determina si se utilizará el primer subespacio 155-1 (espacio de compactación) o el segundo subespacio 155-2 (espacio sin compactación).

El módulo de AMM 150 asigna direcciones lógicas particulares dentro de las porciones 1 a 8 del subespacio 555-2 a los objetos grandes particulares de acuerdo con el tamaño de los objetos. Por ejemplo, el módulo de AMM 150 asigna A2.1.y en la porción 1 a objetos con el tamaño 4KB, asigna A2.1.y en la porción 2 a objetos con el tamaño 8 KB (o menor) y así sucesivamente.

El módulo de AMM 150 le indica a la MMU 130 que proporcione una asignación de las direcciones lógicas asignadas A2.m.y a la memoria física en la RAM 120.

Mientras se ejecuta la aplicación de mutación 140, el módulo de AMM 150 desasignará algunas direcciones lógicas

y reasignará algunas direcciones lógicas (véase la figura 2). Como consecuencia, las porciones 1 a 8 en el subespacio 555-2 se fragmentan. El ejemplo de la figura 5 ilustra un "centro de distribución" en el que las direcciones lógicas ilustradas con relleno negro se asignan a la memoria física P en la RAM 120 (a través de la MMU 130). Por ejemplo, la dirección lógica A2.5.2 (en la porción 5) es asignada a 16 páginas (con 4 KB cada una) de memoria física en la RAM 120 para un objeto particular con un tamaño de 64 KB; la dirección lógica A2.4.8 (en la porción 4) es asignada a la RAM para un objeto particular con un tamaño de 32 KB; hay direcciones lógicas en las porciones 3, 2 y 1 con objetos en la RAM.

No es necesario tener una asignación de uno a uno de las direcciones lógicas a la memoria física. En el ejemplo de la figura 5, la mayor parte de las direcciones lógicas no se asignan a la memoria física.

Se observa que la memoria física utilizada para un solo objeto puede ser contigua o no contigua. Por ejemplo, la dirección lógica A2.2.12 (en la porción 2) con un objeto de 8 KB requeriría solo 2 páginas (cada 4 KB) de memoria física, y la MMU 130 puede ser asignada a direcciones contiguas (véase el ejemplo de P1 y P2 para 0[1] en la figura 1). Si es necesario, la MMU 130 puede ser asignada a direcciones no contiguas en la RAM, por ejemplo, agrupando páginas físicas individuales para asignaciones más grandes (por ejemplo, 4 páginas para el objeto en A2.4.8). La no contigüidad permanece oculta para la aplicación de mutación.

Los tamaños máximos de objeto (MOS) en la figura 5 se ilustran solo a modo de ejemplo. El esquema de potencia de 2 se puede complementar con un MOS intermedio, tal como 6 KB (entre 4 KB y 8 KB), 12 KB (entre 8 KB y 16 KB), 24 KB, etc., hasta 128 GB.

Es posible utilizar otros tamaños de página (PS), por ejemplo, hasta un tamaño de página de 1 GB.

La figura 6 ilustra un ejemplo de un dispositivo informático genérico 900 y un dispositivo informático móvil genérico 950, que puede ser utilizado con las técnicas descritas en el presente documento. El dispositivo informático 900 está destinado a representar diversas formas de ordenadores digitales, tales como ordenadores portátiles, ordenadores de escritorio, estaciones de trabajo, asistentes digitales personales, servidores, servidores de hoja, ordenadores centrales y otros ordenadores apropiados. El dispositivo informático genérico 900 puede corresponder al sistema informático 100 de la figura 1) El dispositivo informático 950 está destinado a representar diversas formas de dispositivos móviles, tales como asistentes digitales personales, teléfonos celulares, teléfonos inteligentes y otros dispositivos informáticos similares. Los componentes que se muestran en el presente documento, sus conexiones y relaciones, y sus funciones, están destinados a ser solo a modo de ejemplo, y no están destinados a limitar la implementación de las invenciones descritas y/o reivindicadas en este documento.

El dispositivo informático 900 incluye un procesador 902, una memoria 904, un dispositivo de almacenamiento 906, una interfaz de alta velocidad 908 que se conecta a la memoria 904 y a los puertos de expansión de alta velocidad 910, y una interfaz de baja velocidad 912 que se conecta al bus de baja velocidad 914 y al dispositivo de almacenamiento 906. Cada uno de los componentes 902, 904, 906, 908, 910 y 912, están interconectados utilizando diversos buses, y pueden estar montados en una placa base común o de otras maneras, según corresponda. El procesador 902 puede procesar instrucciones para la ejecución dentro del dispositivo informático 900, incluidas las instrucciones almacenadas en la memoria 904 o en el dispositivo de almacenamiento 906 para mostrar información gráfica para una GUI en un dispositivo externo de entrada / salida, tal como la pantalla 916 acoplada a la interfaz de alta velocidad 908. En otras implementaciones, se pueden utilizar múltiples procesadores y/o múltiples buses, según corresponda, junto con múltiples memorias y tipos de memoria. Además, se pueden conectar múltiples dispositivos informáticos 900, y cada dispositivo proporciona porciones de las operaciones necesarias (por ejemplo, tal como un banco de servidores, un grupo de servidores de hoja o un sistema de múltiples procesadores).

La memoria 904 almacena información dentro del dispositivo informático 900. En una implementación, la memoria 904 es una unidad o unidades de memoria volátiles. En otra implementación, la memoria 904 es una unidad o unidades de memoria no volátiles. La memoria 904 también puede ser otra forma de medio legible por ordenador, tal como un disco magnético u óptico.

El dispositivo de almacenamiento 906 es capaz de proporcionar almacenamiento masivo para el dispositivo informático 900. En una implementación, el dispositivo de almacenamiento 906 puede ser o contener un medio legible por ordenador, tal como un dispositivo de disquete, un dispositivo de disco duro, un dispositivo de disco óptico o un dispositivo de cinta, una memoria rápida u otro dispositivo de memoria de estado sólido similar, o una variedad de dispositivos, incluidos los dispositivos en una red de área de almacenamiento u otras configuraciones. Un producto de programa informático puede estar incorporado de manera tangible en un soporte de información. El producto de programa informático también puede contener instrucciones que, cuando son ejecutadas, llevan a cabo uno o más métodos, tales como los descritos anteriormente. El portador de información es un medio legible por un ordenador o una máquina, tal como la memoria 904, el dispositivo de almacenamiento 906 o la memoria en el procesador 902.

El controlador de alta velocidad 908 gestiona operaciones que requieren un gran ancho de banda para el dispositivo informático 900, mientras que el controlador de baja velocidad 912 gestiona operaciones que requieren un ancho de banda menor. Dicha asignación de funciones es solo a modo de ejemplo. En una implementación, el controlador de

alta velocidad 908 está acoplado a la memoria 904, a la pantalla 916 (por ejemplo, a través de un procesador gráfico o acelerador) y a los puertos de expansión de alta velocidad 910, que pueden aceptar diversas tarjetas de expansión (no mostradas). En la implementación, el controlador de baja velocidad 912 está acoplado al dispositivo de almacenamiento 906 y al puerto de expansión de baja velocidad 914. El puerto de expansión de baja velocidad, que puede incluir diversos puertos de comunicación (por ejemplo, USB, Bluetooth, Ethernet, Ethernet inalámbrico), puede estar acoplado a uno o más dispositivos de entrada / salida, tal como un teclado, un dispositivo señalador, un escáner o un dispositivo de red, tal como un conmutador o encaminador, por ejemplo, a través de un adaptador de red.

El dispositivo informático 900 puede ser implementado en varias formas diferentes, tal como se muestra en la figura. Por ejemplo, puede ser implementado como un servidor estándar 920, o varias veces en un grupo de dichos servidores. También puede ser implementado como parte de un sistema 924 de servidor en bastidor. Además, se puede implementar en un ordenador personal, tal como un ordenador portátil 922. De manera alternativa, los componentes del dispositivo informático 900 pueden ser combinados con otros componentes en un dispositivo móvil (no mostrado), tal como el dispositivo 950. Cada uno de dichos dispositivos puede contener uno o más de los dispositivos informáticos 900, 950, y un sistema completo puede estar formado por múltiples dispositivos informáticos 900, 950 que se comunican entre sí.

El dispositivo informático 950 incluye un procesador 952, una memoria 964, un dispositivo de entrada / salida tal como una pantalla 954, una interfaz de comunicación 966 y un transceptor 968, entre otros componentes. El dispositivo 950 también puede estar provisto de un dispositivo de almacenamiento, tal como un microdrive u otro dispositivo, para proporcionar almacenamiento adicional. Cada uno de los componentes 950, 952, 964, 954, 966 y 968, están interconectados utilizando diversos buses, y varios de los componentes pueden estar montados en una placa base común o de otras maneras, según corresponda.

El procesador 952 puede ejecutar instrucciones dentro del dispositivo informático 950, incluidas las instrucciones almacenadas en la memoria 964. El procesador puede ser implementado como un conjunto de chips que incluyen múltiples procesadores analógicos y digitales separados. El procesador puede proporcionar, por ejemplo, la coordinación de los otros componentes del dispositivo 950, tal como el control de las interfaces de usuario, las aplicaciones ejecutadas por el dispositivo 950 y la comunicación inalámbrica por el dispositivo 950.

El procesador 952 se puede comunicar con un usuario a través de la interfaz de control 958 y de la interfaz de pantalla 956 acoplada a una pantalla 954. La pantalla 954 puede ser, por ejemplo, una pantalla de LCD con TFT (pantalla de cristal líquido con transistor de película delgada - Thin-Film-Transistor Liquid Crystal Display, en inglés) o una pantalla de OLED (diodo orgánico emisor de luz - Organic Light Emitting Diode, en inglés) u otra tecnología de pantalla apropiada. La interfaz de pantalla 956 puede comprender circuitos apropiados para conducir la pantalla 954 para presentar información gráfica y de otro tipo a un usuario. La interfaz de control 958 puede recibir comandos de un usuario y convertirlos para enviarlos al procesador 952. Además, se puede proporcionar una interfaz externa 962 en comunicación con el procesador 952, para permitir la comunicación de área cercana del dispositivo 950 con otros dispositivos. La interfaz externa 962 puede proporcionar, por ejemplo, comunicación por cable en algunas implementaciones, o comunicación inalámbrica en otras implementaciones, y también se pueden utilizar múltiples interfaces.

La memoria 964 almacena información dentro del dispositivo informático 950. La memoria 964 puede ser implementada como uno o más medios legibles por ordenador, una unidad o unidades de memoria volátil, o una unidad o unidades de memoria no volátil. La memoria de expansión 984 también puede ser proporcionada y conectada al dispositivo 950 a través de la interfaz de expansión 982, que puede incluir, por ejemplo, una interfaz de tarjeta SIMM (Módulo de memoria en línea individual). Dicha memoria de expansión 984 puede proporcionar un espacio de almacenamiento adicional para el dispositivo 950, o también puede almacenar aplicaciones u otra información para el dispositivo 950. De manera específica, la memoria de expansión 984 puede incluir instrucciones para llevar a cabo o complementar los procesos descritos anteriormente, y también puede incluir información segura. De este modo, por ejemplo, la memoria de expansión 984 puede actuar como un módulo de seguridad para el dispositivo 950, y puede ser programada con instrucciones que permitan la utilización segura del dispositivo 950. Además, se pueden proporcionar aplicaciones seguras a través de las tarjetas SIMM, junto con información adicional, tales como colocar la información de identificación en la tarjeta SIMM de manera no pirateable.

La memoria puede incluir, por ejemplo, una memoria rápida y/o una memoria NVRAM, tal como se explica a continuación. En una implementación, un producto de programa informático se incorpora de manera tangible en un soporte de información. El producto de programa informático contiene instrucciones que, cuando son ejecutadas, llevan a cabo uno o más métodos, tales como los descritos anteriormente. El portador de información es un medio legible por un ordenador o una máquina, tal como la memoria 964, la memoria de expansión 984 o la memoria del procesador 952, que puede ser recibida, por ejemplo, a través del transceptor 968 o la interfaz externa 962.

El dispositivo 950 puede comunicarse de forma inalámbrica a través de la interfaz de comunicación 966, que puede incluir circuitos de procesamiento de señal digital cuando sea necesario. La interfaz de comunicación 966 puede proporcionar comunicaciones bajo diversos modos o protocolos, tales como llamadas de voz GSM, SMS, EMS o mensajes MMS, CDMA, TD-MA, PDC, WCDMA, CDMA2000 o GPRS, entre otros. Dicha comunicación puede

ocurrir, por ejemplo, a través del transceptor de radiofrecuencia 968. Además, puede ocurrir una comunicación de corto alcance, tal como la utilización de un Bluetooth, WiFi u otro transceptor (no mostrado). Además, el módulo receptor GPS (Sistema de posicionamiento global - Global Positioning System, en inglés) 980 puede proporcionar datos inalámbricos adicionales relacionados con la navegación y la ubicación al dispositivo 950, que pueden ser utilizados según corresponda por las aplicaciones que se ejecutan en el dispositivo 950.

El dispositivo 950 también puede comunicarse de manera audible utilizando el códec de audio 960, que puede recibir información hablada de un usuario y convertirla en información digital utilizable. El códec de audio 960 también puede generar un sonido audible para un usuario, tal como a través de un altavoz, por ejemplo, en un teléfono del dispositivo 950. Dicho sonido puede incluir sonido de llamadas telefónicas de voz, puede incluir sonido grabado (por ejemplo, mensajes de voz, archivos de música, etc.) y también puede incluir sonido generado por aplicaciones que funcionan en el dispositivo 950.

El dispositivo informático 950 puede ser implementado en varias formas diferentes, tal como se muestra en la figura. Por ejemplo, puede ser implementado como un teléfono celular 980. También puede ser implementado como parte de un teléfono inteligente 982, un asistente digital personal u otro dispositivo móvil similar.

Se pueden realizar diversas implementaciones de los sistemas y técnicas descritos en el presente documento en circuitos electrónicos digitales, circuitos integrados, ASIC especialmente diseñados (circuitos integrados específicos para una aplicación - Application Specific Integrated Circuits, en inglés), hardware, firmware, software y/o combinaciones de los mismos. Estas diversas implementaciones pueden incluir la implementación en uno o más programas informáticos que son ejecutables y/o interpretables en un sistema programable que incluye, al menos, un procesador programable, que puede tener un propósito especial o general, acoplado para recibir datos e instrucciones desde un sistema de almacenamiento, y para transmitir datos e instrucciones al mismo, al menos un dispositivo de entrada y al menos un dispositivo de salida.

Estos programas informáticos (también conocidos como programas, software, aplicaciones de software o código) incluyen instrucciones de la máquina para un procesador programable, y pueden ser implementados en un lenguaje de programación de alto nivel orientado a objetos y/o procedimientos y/o en lenguaje de ensamblador / máquina. Tal como se utiliza en el presente documento, los términos "medio legible por máquina" "medio legible por ordenador" hacen referencia a cualquier producto, aparato y/o dispositivo de programa informático (por ejemplo, discos magnéticos, discos ópticos, memoria, dispositivos lógicos programables (PLD - Programmable Logic Devices, en inglés)) utilizados para proporcionar instrucciones y/o datos de la máquina a un procesador programable, incluido un medio legible por máquina que recibe instrucciones de la máquina como una señal legible por la máquina. El término "señal legible por máquina" hace referencia a cualquier señal utilizada para proporcionar instrucciones de máquina y/o datos a un procesador programable.

Para proporcionar interacción con un usuario, los sistemas y técnicas descritos en el presente documento pueden ser implementados en un ordenador que tenga un dispositivo de visualización (por ejemplo, un CRT (tubo de rayos catódicos - Cathode Ray Tube, en inglés) o un monitor LCD (pantalla de cristal líquido) para mostrar información al usuario y un teclado y un dispositivo señalador (por ejemplo, un ratón o una bola de seguimiento) mediante el cual el usuario puede proporcionar información al ordenador. También se pueden utilizar otros tipos de dispositivos para proporcionar interacción con un usuario; por ejemplo, la retroalimentación proporcionada al usuario puede ser cualquier forma de retroalimentación sensorial (por ejemplo, retroalimentación visual, retroalimentación auditiva o retroalimentación táctil); y la entrada del usuario puede ser recibida de cualquier forma, incluida la entrada acústica, de voz o táctil.

Los sistemas y técnicas descritos en el presente documento pueden ser implementados en un dispositivo informático que incluye un componente del lado del servidor (por ejemplo, tal como un servidor de datos), o que incluye un componente de software intermedio (por ejemplo, un servidor de aplicaciones), o que incluye un componente del lado del usuario (por ejemplo, un ordenador de cliente que tiene una interfaz gráfica de usuario o un navegador web a través del cual un usuario puede interactuar con una implementación de los sistemas y técnicas descritos en el presente documento), o cualquier combinación de dichos componentes del lado del servidor, software intermedio o del lado del usuario. Los componentes del sistema pueden estar interconectados mediante cualquier forma o medio de comunicación de datos digitales (por ejemplo, una red de comunicación). Ejemplos de redes de comunicación incluyen una red de área local ("LAN» - Local Area Network, en inglés), una red de área amplia ("WAN» - Wide Area Network, en inglés) e Internet.

El dispositivo informático puede incluir clientes y servidores. Un cliente y un servidor, en general, están alejados uno de otro y, habitualmente, interactúan a través de una red de comunicación. La relación de cliente y el servidor surge en virtud de los programas informáticos que se ejecutan en los respectivos ordenadores y que tienen una relación de cliente a servidor entre uno y otro.

Se han descrito varias realizaciones. No obstante, se comprenderá que se pueden realizar diversas modificaciones sin apartarse del espíritu y alcance de la invención.

Además, los flujos lógicos representados en las figuras no requieren el orden particular mostrado, o el orden

secuencial, para lograr resultados deseables. Además, se pueden proporcionar otras etapas, o se pueden eliminar etapas, de los flujos descritos, y se pueden agregar o eliminar otros componentes de los sistemas descritos. Por consiguiente, otras realizaciones están dentro del alcance de las siguientes reivindicaciones.

REIVINDICACIONES

1. Método implementado por ordenador (400), para actuar sobre un módulo automático de gestión de memoria (150) en un sistema informático (100) que tiene una memoria de acceso aleatorio (120), comprendiendo el método (400):

para un espacio de direcciones lógicas (155) con un primer subespacio (155-1) y con un segundo subespacio (155-2, 555-2), reservar (410) el segundo subespacio del espacio de direcciones lógicas con porciones que tienen ranuras para objetos de diferentes clases de tamaño, en el que la reserva (410) incluye calcular el tamaño de una porción (1, 2 ... 8) en correspondencia con un tamaño de almacenamiento dinámico (HS) máximo de una aplicación de mutación (140), que es el tamaño acumulado de los objetos que. potencialmente. se almacenarán en el segundo subespacio (155-2, 555-2);

recibir (420), desde la aplicación de mutación (140), solicitudes de asignación de memoria (141) para que objetos particulares (0[1] ... 0[9]) se almacenen en una memoria de acceso aleatorio (120) del sistema informático (100);

asignar (430) direcciones lógicas particulares (Ax.y; A2.1 ... A1.4) dentro de un espacio de direcciones lógicas (155) a los objetos particulares (0[1] ... 0[9]), en el que el módulo de gestión automática de memoria (150) distingue (435) los objetos particulares (0[1] ... 0[9]) de acuerdo con al menos un criterio (TS, OA), y asigna (431, 432) direcciones lógicas (A1.y) de un primer subespacio (155-1) y direcciones lógicas (A2.y) de un segundo subespacio (155-2);

asignar (442) las direcciones lógicas asignadas (A2.y) desde el segundo subespacio (155-2) a la memoria física (P1 ... P104) en la memoria de acceso aleatorio (120) por una unidad de gestión de memoria (130); y

compactar (451) direcciones lógicas dentro del primer subespacio (155-1) en combinación con mover objetos correspondientes en la memoria de acceso aleatorio.

2. El método (400) de acuerdo con la reivindicación 1, en el que en la asignación (430), el módulo de gestión automática de memoria (150) distingue (425) los objetos particulares (0[1] ... 0[9]) de acuerdo con el criterio de que el al menos un (TS, OA) sea un criterio de tamaño de objeto.

3. El método (400) de acuerdo con la reivindicación 2, en el que en la asignación (430), el módulo de gestión automática de memoria (150) utiliza un tamaño umbral (TS) predefinido como criterio de tamaño de objeto y asigna (431) direcciones lógicas en el primer subespacio (155-1) a objetos que tienen un tamaño de objeto por debajo del tamaño umbral (TS) predefinido, y asigna (432) direcciones lógicas en el segundo subespacio (155-2) a objetos que tienen un tamaño de objeto por encima del tamaño umbral (TS) predefinido o igual al tamaño umbral predefinido (TS).

4. El método (400) de acuerdo con la reivindicación 3, en el que el tamaño umbral predefinido (TS) está relacionado con un tamaño de página (PS) de la memoria de acceso aleatorio (120).

5. El método (400) de acuerdo con la reivindicación 4, en el que el tamaño umbral (TS) predefinido es igual al tamaño de página (PS) o a un múltiplo del tamaño de página.

6. El método (400) de acuerdo con la reivindicación 1, en el que la reserva (410) incluye reservar porciones (1, 2 ...), en el que una porción de inicio tiene un primer número de ranuras como un número de potencia de 2, y las ranuras adyacentes tienen números de ranuras calculadas en una serie geométrica a partir del primer número de ranuras.

7. El método (400) de acuerdo con cualquiera de las reivindicaciones anteriores, con las etapas adicionales de desasignar algunas de las direcciones lógicas particulares de cualquiera de los subespacios (155-1, 155-2), dando instrucciones con ello al módulo de gestión de memoria (130) para desasignar la memoria física de las direcciones lógicas asignadas previamente en el segundo subespacio (155-2) sin compactar los objetos en la memoria de acceso aleatorio (120).

8. El método (400) de acuerdo con la reivindicación 1, en el que el módulo de gestión automática de memoria (150) distingue (425) los objetos particulares (0[1] ... 0[9]) de acuerdo con que al menos un criterio sea un atributo de objeto (OA).

9. El método (400) de acuerdo con la reivindicación 8, en el que el atributo de objeto (OA) comprende atributos seleccionados entre los siguientes: tipo de objeto, frecuencia de acceso a objetos, el objeto que se pasa a código externo y el hardware al que accede directamente el objeto.

10. Un sistema informático (100) para llevar a cabo las etapas del método implementado en el ordenador, de acuerdo con cualquiera de las reivindicaciones 1 a 9.

11. Un producto de programa informático que, cuando está cargado en la memoria de un ordenador y es

ejecutado por al menos un procesador del ordenador, lleva a cabo las etapas del método implementado por el ordenador, de acuerdo con cualquiera de las reivindicaciones 1 a 9.

12. El producto de programa informático de acuerdo con la reivindicación 11, en el que el módulo de gestión automática de memoria (150) forma parte de un entorno de tiempo de ejecución (RTE) para aplicaciones de mutación (140, 140') escrito en un lenguaje de programación seleccionado del grupo de: JAVA, C#, PYTHON, JAVASCRIPT, SCALA y GO.

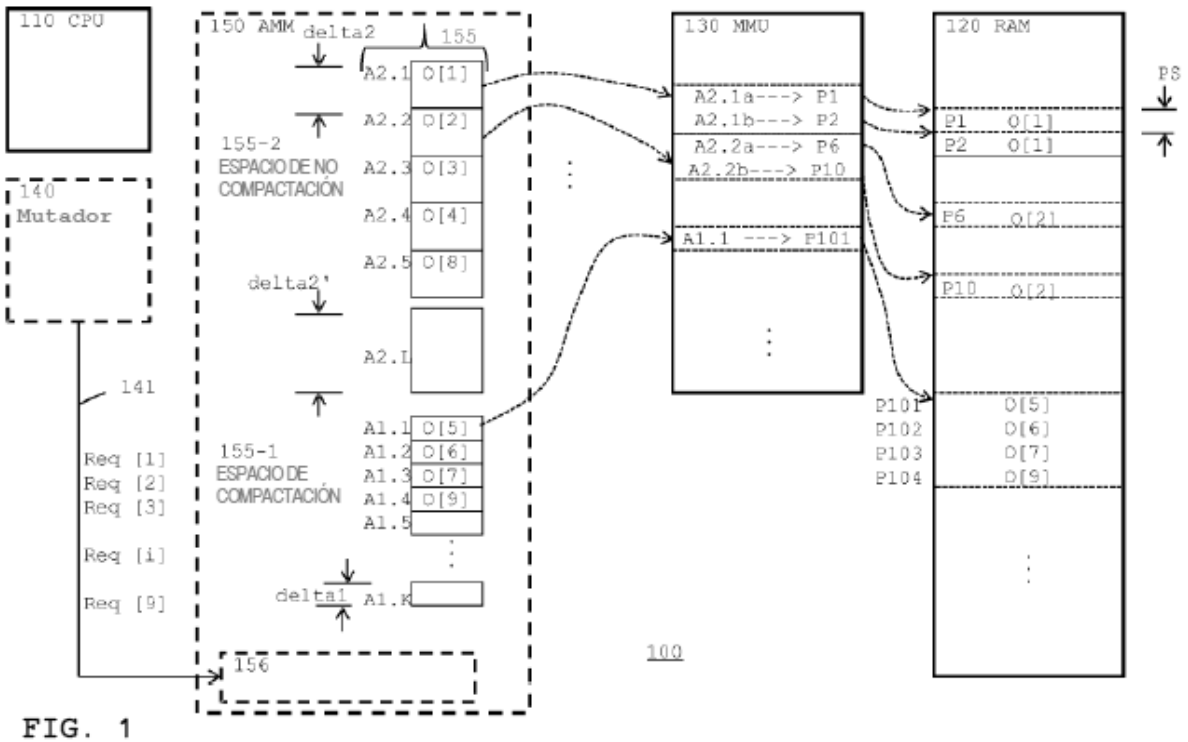
110 CPU
140
Mutador
141
Req [1]
Req [2]
Req [3]
Req [i]
Req [9]
150 AMM
delta2
155
A2.1 O[1]
A2.2 O[2]
155-2
ESPACIO DE NO
COMPACTACIÓN
A2.3 O[3]
A2.4 O[4]
A2.5 O[8]
delta2'
A2.L
A1.1 O[5]
A1.2 O[6]
155-1
ESPACIO DE
COMPACTACIÓN
A1.3 O[7]
A1.4 O[9]
A1.5
delta1
A1.K
156
130 MMU
A2.1a---> P1
A2.1b---> P2
A2.2a---> P6
A2.2b---> P10
A1.1 ---> P101
120 RAM
PS
P1 O[1]
P2 O[1]
P6 O[2]
P10 O[2]
P101 O[5]
P102 O[6]
P103 O[7]
P104 O[9]
100

FIG. 1

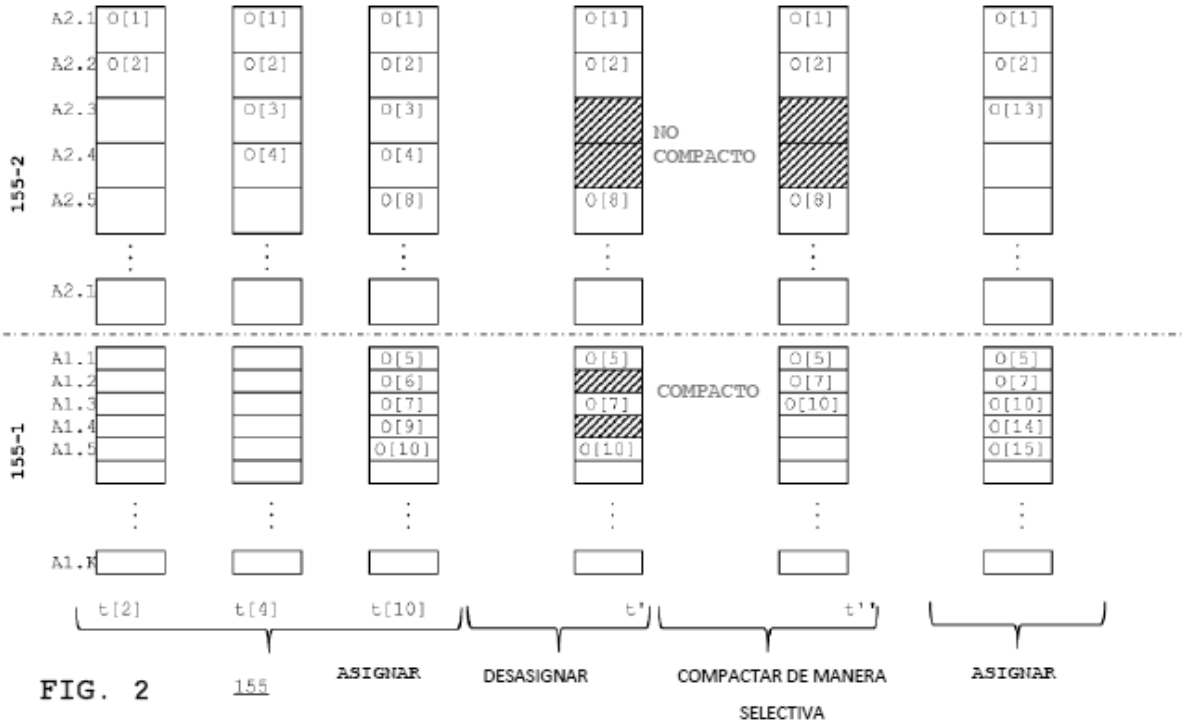
155-2
155-1
A2.1
A2.2
A2.3
A2.4
A2.5
A2.I
A1.1
A1.2
A1.3
A1.4
A1.5
A1.N
O[1]
O[2]
O[3]
O[4]
O[8]
O[13]
O[5]
O[6]
O[7]
O[9]
O[10]
O[14]
O[15]
NO COMPACTO
COMPACTO
t[2]
t[4]
t[10]
t'
t''
ASIGNAR
DESASIGNAR
COMPACTAR DE MANERA SELECTIVA
ASIGNAR

FIG. 2 155

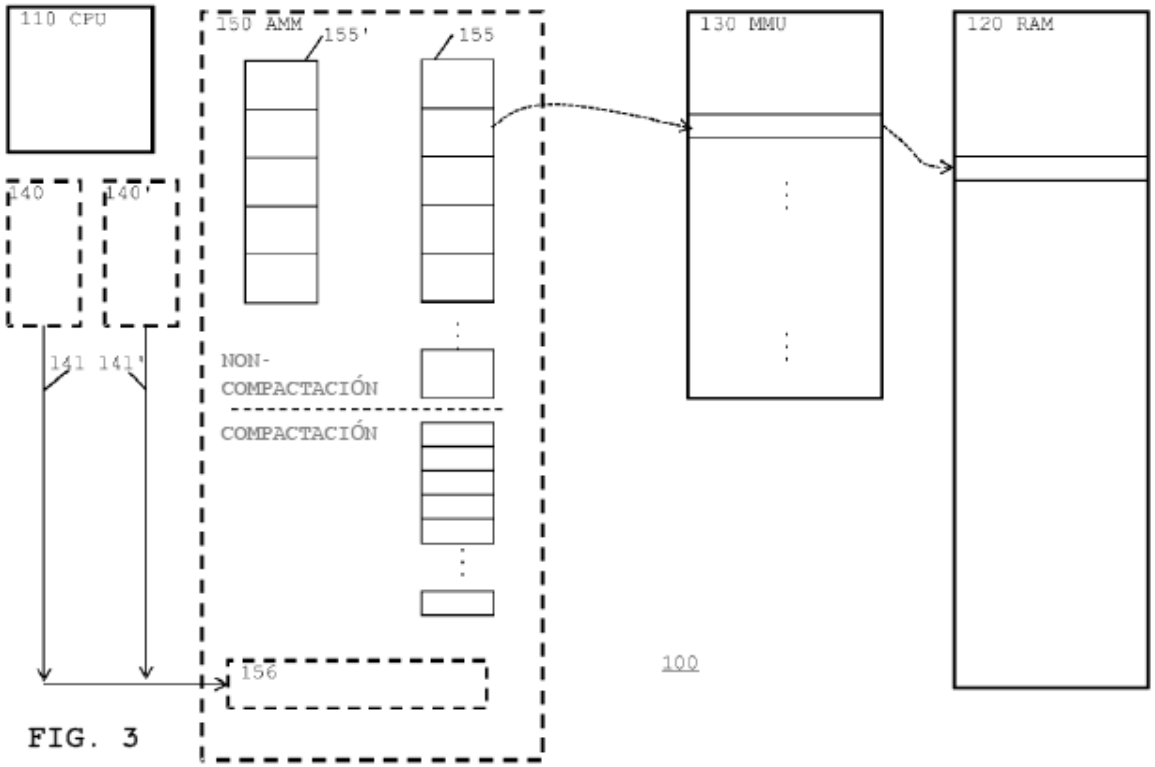
110 CPU
150 AMM
155'
155
130 MMU
120 RAM
140
140'
141
141'
NON-
COMPACTACIÓN
COMPACTACIÓN
156
100

FIG. 3

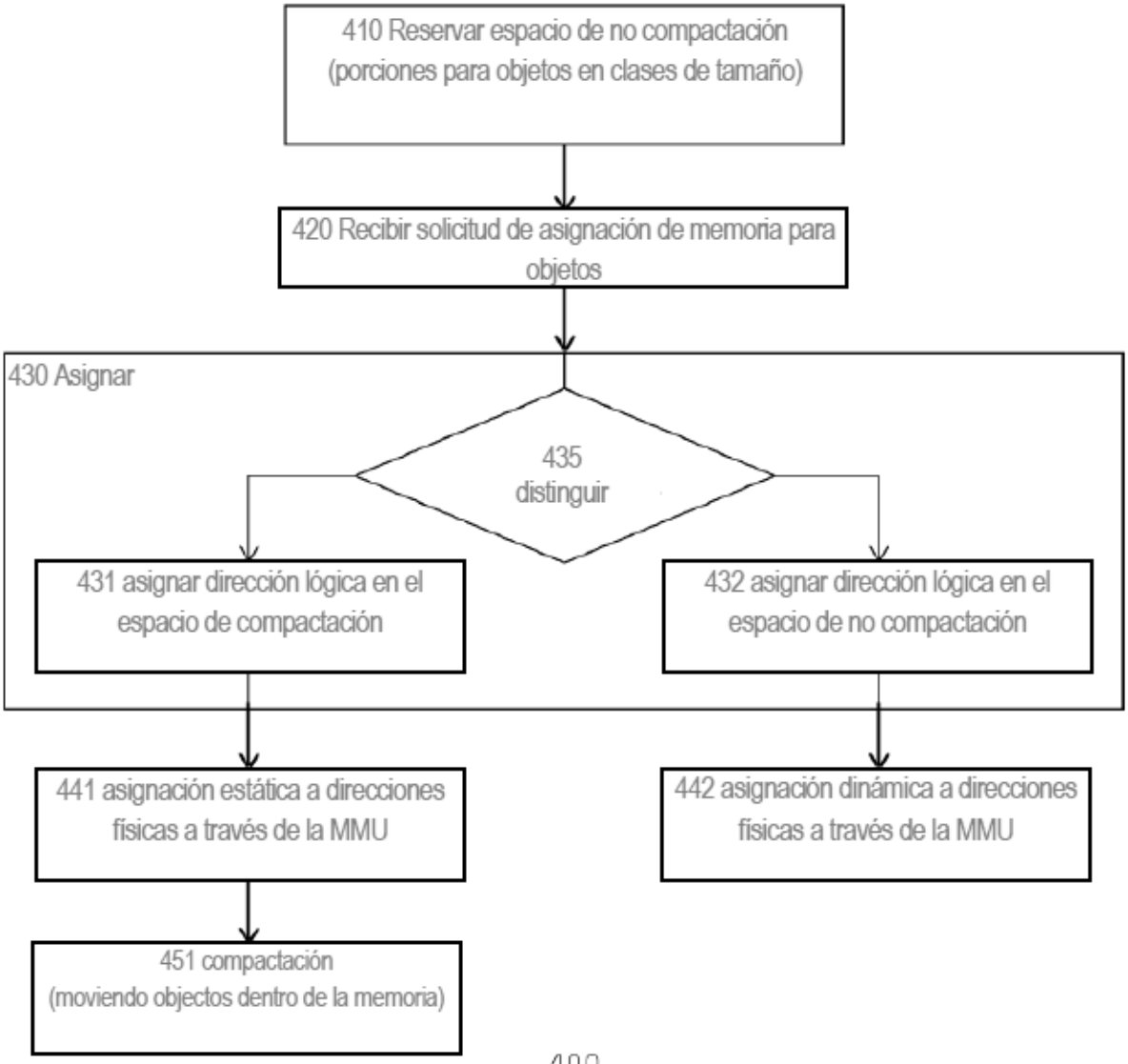
410 Reservar espacio de no compactación
(porciones para objetos en clases de tamaño)
420 Recibir solicitud de asignación de memoria para objetos
430 Asignar
435
distinguir
431 asignar dirección lógica en el espacio de compactación
432 asignar dirección lógica en el espacio de no compactación
441 asignación estática a direcciones físicas a través de la MMU
442 asignación dinámica a direcciones físicas a través de la MMU
451 compactación
(moviendo objectos dentro de la memoria)
400

FIG. 4

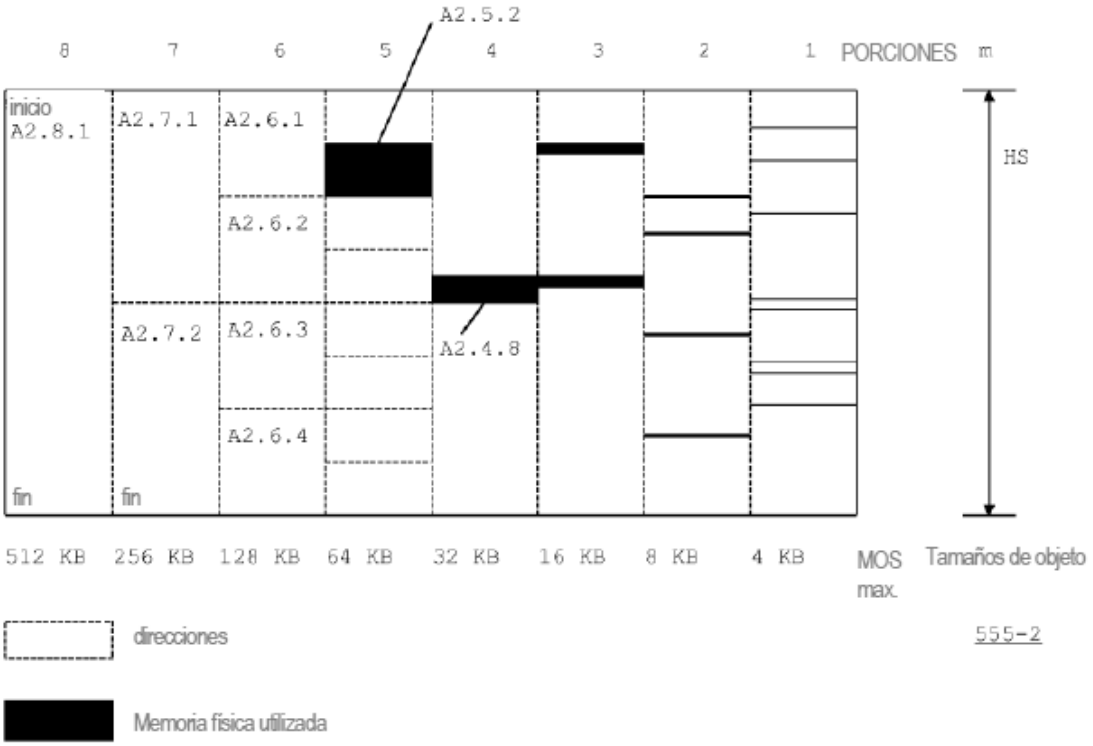
A2.5.2
8
7
6
5
4
3
2
1
PORCIONES
m
inicio
A2.8.1
A2.7.1
A2.6.1
A2.6.2
A2.7.2
A2.6.3
A2.4.8
A2.6.4
fin
fin
HS
512 KB
256 KB
128 KB
64 KB
32 KB
16 KB
8 KB
4 KB
MOS
max.
Tamaños de objeto
direcciones
555-2
Memoria física utilizada

FIG. 5

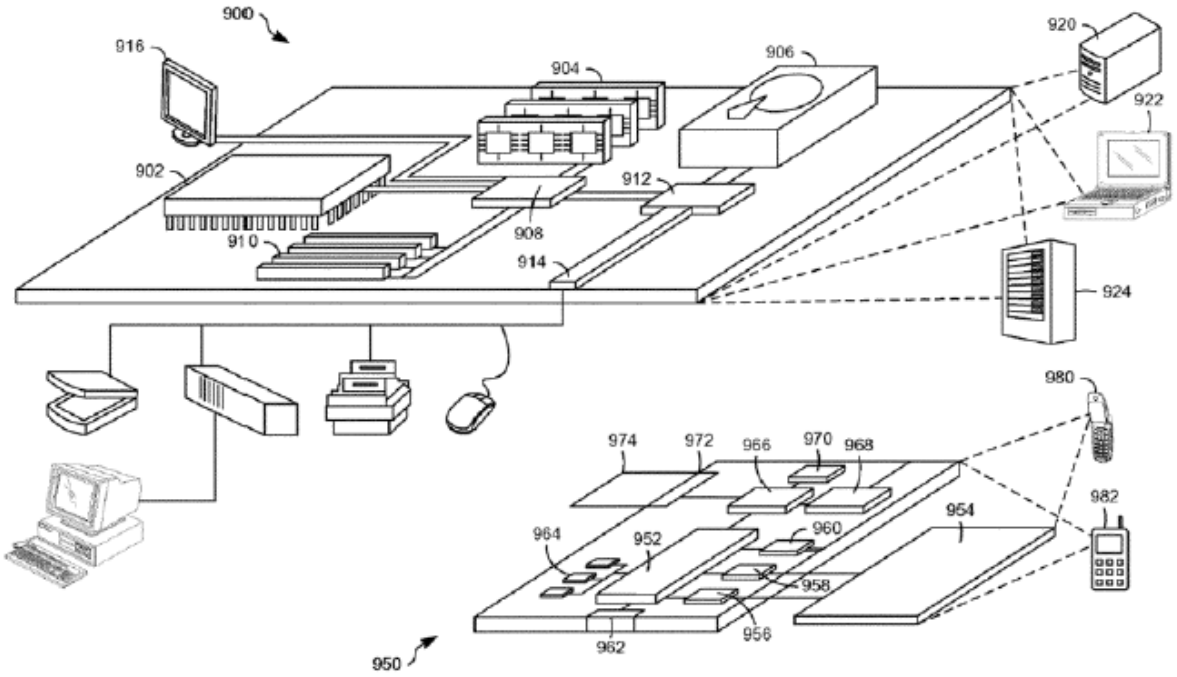
900
916
904
906
920
922
902
912
908
910
914
924
980
974
972
966
970
968
982
964
952
954
960
958
956
962
950

FIG. 6